



OCPP 2.0.1
Part 6 - Test Cases

Edition 4, 2025-12-03

Table of Contents

| | |
|--|-----|
| 1. Introduction | 2 |
| About this document | 2 |
| Conventions | 2 |
| 2. Test Cases Charging Station | 3 |
| A Security | 4 |
| B Provisioning | 28 |
| C Authorization | 91 |
| D Local Authorization List Management | 164 |
| E Transactions | 173 |
| F Remote Control | 230 |
| G Availability | 255 |
| H Reservation | 280 |
| I Tariff and Cost | 309 |
| J Meter Values | 314 |
| K Smart Charging | 326 |
| L Firmware Management | 389 |
| M CertificateManagement | 429 |
| N Diagnostics | 455 |
| O Display Message | 528 |
| P Data Transfer | 572 |
| Memory states | 575 |
| Reusable states | 584 |
| 3. Test Cases Charging Station Management System | 608 |
| A Security | 609 |
| B Provisioning | 629 |
| C Authorization | 655 |
| D Local Authorization List Management | 674 |
| E Transactions | 680 |
| F Remote Control | 711 |
| G Availability | 727 |
| H Reservation | 737 |
| I Tariff and Cost | 750 |
| J Meter Values | 754 |
| K Smart Charging | 764 |
| L Firmware Management | 800 |
| M Certificate Management | 826 |
| N Diagnostics | 844 |
| O Display Message | 878 |
| P Data Transfer | 899 |
| Reusable states | 901 |

Copyright © 2010 - 2025 Open Charge Alliance. All rights reserved.

This document is made available under the **Creative Commons Attribution-NoDerivatives 4.0 International Public License** (<https://creativecommons.org/licenses/by-nd/4.0/legalcode>).

Version History

| Version | Reviewed by | Modified by | Description |
|----------------------|-------------|----------------------|---|
| OCPP 2.0.1 Edition 4 | 2025-12-03 | Open Charge Alliance | OCPP 2.0.1 Edition 4. All errata from OCPP 2.0.1 Part 6 until and including Errata 2025-11 have been merged into this version of the specification. |
| OCPP 2.0.1 Edition 3 | 2024-05-06 | Open Charge Alliance | OCPP 2.0.1 Edition 3. All errata from OCPP 2.0.1 Part 6 until and including Errata 2024-04 have been merged into this version of the specification. In this edition all certification profiles are available. |
| OCPP 2.0.1 | 2023-06-30 | Open Charge Alliance | Release for Core & Advanced Security |

1. Introduction

About this document

This document is created to describe a set of valid test cases for OCPP 2.0.1. These test cases can be executed using a Test System for OCPP 2.0.1. The scenarios in the tool are described in detail including the expected behaviour of the System Under Test (SUT). This document is divided in chapters, each describing an OCPP functional block as can be found in the official OCPP specification. These are:

- A. Security
- B. Provisioning
- C. Authorization
- D. Local Authorization List Management
- E. Transactions
- F. Remote Control
- G. Availability
- H. Reservation
- I. Tariff and Cost
- J. Meter Values
- K. Smart Charging
- L. Firmware Management
- M. Certificate Management
- N. Diagnostics
- O. Display Message
- P. Data Transfer

The scenarios in this document are also part of the OCA certification process of OCPP. Please refer to OCPP 2.0.1 Part 5 - Certification Profiles for more information about the relation between certification profiles and the test scenarios in this document.

Conventions

The following conventions / rules apply to all test cases, unless explicitly mentioned otherwise. These will not be mentioned separately at every test case.

- The OCPP specification is always leading.
- This document does not specify which tests need to be passed for certification, this will be specified in a separate document.
- All messages shall comply with the OCPP 2.0.1 schemas from the OCPP specification.
- The messages are to be sent as mentioned in the scenario details.
- Validations will be mentioned and grouped per step.
- Messages, datatypes and configuration variables will convey to the following formatting rules:
 - Datatypes, messages and configuration variables are displayed bold.
 - Values are displayed italic.

2. Test Cases Charging Station

General pre conditions & tool validations

General conditions/validations are overruled by testcase specific conditions/validations, unless specifically stated otherwise.

General pre conditions:

- Charging Station is Accepted by the CSMS
- Charging Station has a stable active connection to the CSMS
- Charging Station connectors are available
- Charging Station is Idle, with no active transactions
- Charging Station is clear of faults
- Charging Station has no charging schedules active
- Charging Station has no active reservations
- The Configuration variable **AuthCtrlr.LocalPreAuthorize** is set to *false*.
- Charging Station has no more OCPP messages to be send in queue
- Charging Station is not busy with transfer of diagnostics
- Charging Station is not busy with download of firmware
- Charging Station is not upgrading firmware
- Charging Station is ready to accept/start a charging session
- Charging Station has no Display message configured
- Charging Station has no active custom monitors

General tool rules/validations:

- TransactionEventRequest messages don't have to be sent in chronological order. However the provided seqNo are sequentially numbered in chronological order. This way the CSMS is able to determine whether all messages of a transaction have been received.
- After connecting/disconnecting the EV and EVSE, the Charging Station SHALL report the new status of its connector and report any queued TransactionEventRequest(s). These message are allowed to be sent in any order.
- If the transaction was authorized with **Reusable State** *Authorized* remote, then the first TransactionEventRequest sent after receiving a **RequestStartTransactionRequest** message will contain **triggerReason** with value *_RemoteStart* (This will overrule the step specific tool validations) AND will contain **transactionInfo.remoteStartId**
- The first **TransactionEventRequest** of a transaction MUST contain **eventType** *Started*.
- The first **TransactionEventRequest** sent after connecting the EVSE and EV MUST contain **evse.id** and **evse.connectorId**
- The first **TransactionEventRequest** sent after presenting the idToken MUST contain **idToken** with value *<Configured valid idToken fields>*
- If the energy transfer was stopped with **Reusable State** *StopAuthorized* local, then the **_stoppedReason** of the last **TransactionEventRequest** of that transaction with **eventType** *Ended*, must have value *Local* OR be omitted.
- When validating/comparing time / dateTime values, the Test System will in most cases accept a configurable deviation. The certification labs will configure a deviation of 4 seconds.
- Every FirmwareStatusNotificationRequest sent for a firmware update SHALL contain the same requestId as the UpdateFirmwareRequest that started the firmware update.
- The list of ChargingSchedulePeriod elements in a chargingSchedule SHALL be ordered by increasing values of ChargingSchedulePeriod.startPeriod. This means the list is in chronological order.
- When idToken type *NoAuthorization* is configured to be used, the Test System will act/validate differently. No AuthorizeRequest is expected anymore and the value of the idToken at the TransactionEventRequest should be an empty string *""*. Additionally many testcases like Authorization cache, local authorization list, groupIdToken, etc. Will not work for this idToken type.

A Security

TC_A_01_CS: Basic Authentication - Valid username/password combination

| | |
|--------------------------|---|
| Test case name | Basic Authentication - Valid username/password combination |
| Test case Id | TC_A_01_CS |
| Use case Id(s) | A00, B01 |
| Requirement(s) | A00.FR.202, A00.FR.203, A00.FR.204, A00.FR.205, A00.FR.301, A00.FR.302, A00.FR.304 AND B01.FR.01, B01.FR.05, B01.FR.09 |
| System under test | Charging Station |
| Description | The Charging Station uses Basic authentication to authenticate itself to the CSMS, when using security profile 1 or 2. |
| Purpose | To verify whether the Charging Station is able to authenticate itself to the CSMS using Basic Authentication. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The charging station supports security profile 1 and/or 2 - The active NetworkConnectionProfile uses either security profile 1 OR 2. |

| |
|---|
| Before (Preparations) |
| Configuration State: SecurityCtrlr.BasicAuthPassword is <Configured basicAuthPassword> |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>Booted</i> | |

| |
|---|
| Tool validations |
| <p>* Step 1:</p> <p>The authorization header of the HTTP upgrade request must be formatted as follows: AUTHORIZATION: Basic <Base64 encoded(<ChargingStationId>:<Configured basicAuthPassword>)></p> <ul style="list-style-type: none"> - The ChargingStationId, must equal the ChargingStationId provided at the end of the connection url string of the HTTP request. - BasicAuthPassword must consist of minimum 16 and maximum 40 characters - BasicAuthPassword may only contain alpha-numeric characters and the special characters allowed by passwordString. |
| Post scenario validations: N/a |

TC_A_04_CS: TLS - server-side certificate - Valid certificate

| | |
|-------------------|---|
| Test case name | TLS - server-side certificate - Valid certificate |
| Test case Id | TC_A_04_CS |
| Use case Id(s) | A00 |
| Requirement(s) | A00.FR.309,A00.FR.312,A00.FR.313,A00.FR.319,A00.FR.321,A00.FR.412,A00.FR.422 |
| System under test | Charging Station |
| Description | The CSMS uses a server-side certificate to identify itself to the Charging Station, when using security profile 2 or 3. |
| Purpose | To verify whether the Charging Station is able to receive a server certificate provided by the CSMS and setup a secured WebSocket connection. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The charging station supports security profile 2 and/or 3 - The active NetworkConnectionProfile uses either security profile 2 OR 3. |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>Booting</i> |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Charging Station initiates a TLS handshake and sends a Client Hello to the Test System. | 2. The Test System responds with a Server Hello With the <Configured server certificate> |
| 3. The Charging Station performs the following actions: Send client certificate Client Key Exchange Certificate verify Change Cipher Spec Finished <u>Note(s):</u> - The client certificate is only sent when the Charging Station uses security profile 3. | 4. The Test System performs the following actions: Change Cipher Spec Finished |
| 5. The Charging Station sends a HTTP upgrade request to the Test System <u>Note(s):</u> - The HTTP request only contains a username/password combination when the Charging Station uses security profile 2. | 6. The Test System upgrades the connection to a (secured) WebSocket connection. |
| 7. The Charging Station sends a BootNotificationRequest | 8. The Test System responds with a BootNotificationResponse with status Accepted |
| 9. The Charging Station notifies the CSMS about the current state of all connectors. | 10. The Test System responds accordingly. |

| Tool validations |
|--|
| <p>* Step 2:</p> <p>The Test System validates the following before sending the server certificate:</p> <ul style="list-style-type: none">- The Charging Station must use TLS version 1.2 or above <p>At least the following set of cipher suites must be supported:</p> <p>(TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256</p> <p>AND</p> <p>TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384)</p> <p>OR</p> <p>(TLS_RSA_WITH_AES_128_GCM_SHA256</p> <p>AND</p> <p>TLS_RSA_WITH_AES_256_GCM_SHA384)</p> <p>* Step 9:</p> <p>Message: StatusNotificationRequest</p> <ul style="list-style-type: none">- connectorStatus <i>Available</i> <p>Message: NotifyEventRequest</p> <ul style="list-style-type: none">- eventData[0].trigger <i>Delta</i>- eventData[0].actualValue <i>"Available"</i>- eventData[0].component.name <i>"Connector"</i>- eventData[0].variable.name <i>"AvailabilityState"</i> |
| <p>Post scenario validations:</p> <p>N/a</p> |

TC_A_05_CS: TLS - server-side certificate - Invalid certificate

| | |
|-------------------|---|
| Test case name | TLS - server-side certificate - Invalid certificate |
| Test case Id | TC_A_05_CS |
| Use case Id(s) | A00 |
| Requirement(s) | A00.FR.309,A00.FR.310,A00.FR.311,A00.FR.412,A00.FR.413,A00.FR.414 |
| System under test | Charging Station |
| Description | The CSMS uses a server-side certificate to identify itself to the Charging Station, when using security profile 2 or 3. |
| Purpose | To verify whether the Charging Station is able to terminate the connection when the received server certificate is invalid. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The charging station supports security profile 2 and/or 3 - The active NetworkConnectionProfile uses either security profile 2 OR 3. |

| |
|--|
| Before (Preparations) |
| Configuration State: OCPPCommCtrlr.NetworkProfileConnectionAttempts is 3 OCPPCommCtrlr.NetworkConfigurationPriority only contains <Value from ActiveNetworkProfile> SecurityCtrlr.AllowCSMSTLSWildcards is false (If implemented) |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Test System aborts the connection with the Charging Station. | |
| 2. The Charging Station initiates a TLS handshake and sends a Client Hello to the Test System. | 3. The Test System responds with a Server Hello With a <Configured valid server certificate> <u>Note(s):</u> - The Test System will use this as an indication of the time it takes the Charging Station to reconnect. |
| 4. The Test System aborts the connection with the Charging Station. | |
| 5. The Charging Station initiates a TLS handshake and sends a Client Hello to the Test System. | 6. The Test System responds with a Server Hello With a <Generated invalid server certificate> |
| 7. The Charging Station deems the server certificate invalid and terminates the connection. | |
| <u>Note:</u> The Test System will wait two times the measured reconnection time from step 3, before switching the server certificate back to the valid server certificate. The reason for this is that the Test System is not always able to detect a failed connection attempt. | |
| 8. The Charging Station initiates a TLS handshake and sends a Client Hello to the Test System. | 9. The Test System responds with a Server Hello With a <Configured valid server certificate> <u>Note(s):</u> - The Test System will accept the connection to prevent doubling of the RetryBackOffWaitMinimum. |
| 10 The Charging Station sends a SecurityEventNotificationRequest | 11 The Test System responds with a SecurityEventNotificationResponse |

Main (Test scenario)Note(s):

The Test System will loop through steps 4 to 11 for a set of generated invalid certificates;
"Expired", "Future validity date", "Not signed by installed CSMS Root certificate", "CommonName that does not equal the FQDN of the server", "CommonName containing a wildcard hostname matching the FQDN".

Tool validations

* Step 10:

Message: **SecurityEventNotificationRequest**

- **type** must be *InvalidCsmsCertificate*

Post scenario validations:

N/a

TC_A_06_CS: TLS - server-side certificate - TLS version too low

| | |
|-------------------|---|
| Test case name | TLS - server-side certificate - TLS version too low |
| Test case Id | TC_A_06_CS |
| Use case Id(s) | A00 |
| Requirement(s) | A00.FR.314,A00.FR.316,A00.FR.416,A00.FR.417,A00.FR.419 |
| System under test | Charging Station |
| Description | The CSMS uses a server-side certificate to identify itself to the Charging Station, when using security profile 2 or 3. |
| Purpose | To verify whether the Charging Station is able to terminate the connection when it notices the used TLS version is lower than 1.2. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The charging station supports security profile 2 and/or 3 - The active NetworkConnectionProfile uses either security profile 2 OR 3. |

| |
|--|
| Before (Preparations) |
| Configuration State: OCPPCommCtrlr.NetworkProfileConnectionAttempts is 1 |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Charging Station initiates a TLS handshake and sends a Client Hello to the Test System. | 2. The Test System responds with a Server Hello, but uses a TLS version lower than 1.2 With a <Configured server certificate> |
| 3. The Charging Station notices the used TLS version is lower than 1.2 and terminates the connection. | |
| 4. The Charging Station initiates a TLS handshake and sends a Client Hello to the Test System. | 5. The Test System responds with a Server Hello With the <Configured server certificate> |
| 6. The Charging Station performs the following actions: Send client certificate Client Key Exchange Certificate verify Change Cipher Spec Finished <u>Note(s):</u> - The client certificate is only sent when the Charging Station uses security profile 3. | 7. The Test System performs the following actions: Change Cipher Spec Finished |
| 8. The Charging Station sends a HTTP upgrade request to the Test System <u>Note(s):</u> - The HTTP request only contains a username/password combination when the Charging Station uses security profile 2. | 9. The Test System upgrades the connection to a (secured) WebSocket connection. |
| 10. The Charging Station sends a BootNotificationRequest | 11. The Test System responds with a BootNotificationResponse with status Accepted |
| 12. The Charging Station notifies the CSMS about the current state of all connectors. | 13. The Test System responds accordingly. |

| Main (Test scenario) | |
|---|--|
| 14 The Charging Station sends a SecurityEventNotificationRequest | 15 The Test System responds with a SecurityEventNotificationResponse |
| 16 The Charging Station sends a SecurityEventNotificationRequest | 17 The Test System responds with a SecurityEventNotificationResponse |
| <u>Note(s):</u> - The order in which the requests of steps 12 and 14 and 16 arrive is not relevant. - Steps 16 and 17 are optional as the Charging Station might not be able to detect that the TLS handshake failed, because of invalid TLS version. | |

| Tool validations |
|---|
| * Step 14: Message: SecurityEventNotificationRequest - type must be <i>StartupOfTheDevice</i> or <i>ResetOrReboot</i> |
| * Step 16: Message: SecurityEventNotificationRequest - type must be <i>InvalidTLSVersion</i> |
| Post scenario validations: N/a |

TC_A_07_CS: TLS - Client-side certificate - valid certificate

| | |
|-------------------|--|
| Test case name | TLS - Client-side certificate - valid certificate |
| Test case Id | TC_A_07_CS |
| Use case Id(s) | A00 |
| Requirement(s) | A00.FR.401,A00.FR.402,A00.FR.415,A00.FR.416,A00.FR.422,A00.FR.502,A00.FR.503,A00.FR.507,A00.FR.508,A00.FR.511 |
| System under test | Charging Station |
| Description | The Charging Station uses a client-side certificate to identify itself to the CSMS, when using security profile 3. |
| Purpose | To verify whether the Charging Station is able to provide a valid client certificate and setup a secured WebSocket connection. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The charging station supports security profile 3 - The active NetworkConnectionProfile uses security profile 3. |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State(s):

State is *Booting*

Main (Test scenario)

| Charging Station | CSMS |
|---|--|
| 1. The Charging Station initiates a TLS handshake and sends a Client Hello to the Test System. | 2. The Test System responds with a Server Hello With the <Configured server certificate> |
| 3. The Charging Station performs the following actions: Send client certificate Client Key Exchange Certificate verify Change Cipher Spec Finished | 4. The Test System performs the following actions: Change Cipher Spec Finished |
| 5. The Charging Station sends a HTTP upgrade request to the Test System | 6. The Test System upgrades the connection to a (secured) WebSocket connection. |
| 7. The Charging Station sends a BootNotificationRequest | 8. The Test System responds with a BootNotificationResponse with status Accepted |
| 9. The Charging Station notifies the CSMS about the current state of all connectors. | 10. The Test System responds accordingly. |

Tool validations*** Step 4:**

The Test System validates the following before finishing the TLS handshake:

- The Charging Station must use TLS version 1.2 or above

At least the following set of cipher suites must be supported:

(TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256

AND

TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384)

OR

(TLS_RSA_WITH_AES_128_GCM_SHA256

AND

TLS_RSA_WITH_AES_256_GCM_SHA384)

- When using RSA or DSA the key must be at least 2048 bits long.

and when using elliptic curve cryptography the key must be at least 224 bits long.

- The received Client side certificate must be transmitted in the X.509 format encoded in Privacy-Enhanced Mail (PEM) format.

- The certificate must include a serial number.

- The subject field of the certificate must contain a commonName RDN which consists of the unique serial number of the Charging Station.

NOTE: If one of the above validations fails, the Test System can still setup the WebSocket connection (if it is able to), but the testcase will FAIL and the Test System reports why it failed.

*** Step 9:**

Message: **StatusNotificationRequest**

- **connectorStatus** *Available*

Message: **NotifyEventRequest**

- **eventData[0].trigger** *Delta*

- **eventData[0].actualValue** *"Available"*

- **eventData[0].component.name** *"Connector"*

- **eventData[0].variable.name** *"AvailabilityState"*

Post scenario validations:

N/a

TC_A_09_CS: Update Charging Station Password for HTTP Basic Authentication - Accepted

| | |
|-------------------|--|
| Test case name | Update Charging Station Password for HTTP Basic Authentication - Accepted |
| Test case Id | TC_A_09_CS |
| Use case Id(s) | A01 |
| Requirement(s) | A01.FR.01, A01.FR.11, A01.FR.12, B01.FR.01 |
| System under test | Charging Station |
| Description | This test case defines how to use the BasicAuthPassword, the password used to authenticate Charging Stations in security profile 1 (Basic Authentication) and security profile 2 (TLS with Basic Authentication) |
| Purpose | To verify if the Charging Station is able to accept and store and log the new BasicAuthPassword as described at the OCPP specification. |
| Prerequisite(s) | The charging station supports security profile 1 and/or 2 |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a SetVariablesResponse | 1. The Test System sends a SetVariablesRequest with setVariableData[1]: - variable.name = "BasicAuthPassword" - component.name = "SecurityCtrlr" - attributeValue = "<NewPassword>" |
| 3. The ChargingStation sends a HTTP upgrade request with an Authorization header, containing a username/password combination (with the new <i>BasicAuthPassword</i>). <u>Note(s):</u> - The Authorization header is formatted as follows: <i>AUTHORIZATION: Basic <Base64 encoded(<Configured ChargingStationId>:<NEW BasicAuthPassword>)></i> | 4. The Test System validates the username/password combination AND upgrades the connection to a (secured) WebSocket connection. |
| 5. The Charging Station sends a BootNotificationRequest | 6. The Test System responds with a BootNotificationResponse |
| 7. The Charging Station notifies the Test System about the current state of all connectors. | 8. The Test System responds accordingly. |
| <u>Note(s):</u> - Steps 5, 6, 7, and 8 are only required when status in Step 2 is <i>RebootRequired</i> | |

| |
|---|
| Tool validations |
| * Step 2: Message: SetVariablesResponse - status must be <i>Accepted</i> or <i>RebootRequired</i> |
| Post scenario validations: N/a |

TC_A_10_CS: Update Charging Station Password for HTTP Basic Authentication - Rejected

| | |
|-------------------|--|
| Test case name | Update Charging Station Password for HTTP Basic Authentication - Rejected |
| Test case Id | TC_A_10_CS |
| Use case Id(s) | A01 |
| Requirement(s) | A01.FR.01, A01.FR.11, A01.FR.12 |
| System under test | Charging Station |
| Description | This test case defines how to use the BasicAuthPassword, the password used to authenticate Charging Stations in security profile 1 (Basic Authentication) and security profile 2 (TLS with Basic Authentication) |
| Purpose | To verify if the Charging Station is able to reject the new BasicAuthPassword. |
| Prerequisite(s) | The charging station supports security profile 1 and/or 2 |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a SetVariablesResponse | <p>1. The Test System sends a SetVariablesRequest</p> <p>setVariableData[1]:</p> <ul style="list-style-type: none"> - variable.name = "BasicAuthPassword" - component.name = "SecurityCtrlr" - attributeValue = "<NewPassword which is less than 16 characters>" |
| | <p>3. The ChargingStation sends a HTTP upgrade request with an Authorization header, containing a username/password combination (with the old <i>BasicAuthPassword</i>).</p> <p><u>Note(s):</u></p> <ul style="list-style-type: none"> - The Authorization header is formatted as follows: AUTHORIZATION: Basic <Base64 encoded(<Configured ChargingStationId>:<OLD BasicAuthPassword>)> |
| 4. The Test System validates the username/password combination AND upgrades the connection to a (secured) WebSocket connection. | |
| 5. Execute Reusable State <i>Booted</i> | |

| |
|--|
| Tool validations |
| <p>* Step 2:</p> <p>Message: SetVariablesResponse</p> <ul style="list-style-type: none"> - status must be <i>Rejected</i> |
| <p>Post scenario validations:</p> <p>BasicAuthPassword should be <Configured BasicAuthPassword></p> <p>N/a</p> |

TC_A_11_CS: Update Charging Station Certificate by request of CSMS - Success - Charging Station Certificate

| | |
|--------------------------|--|
| Test case name | Update Charging Station Certificate by request of CSMS - Success - Charging Station Certificate |
| Test case Id | TC_A_11_CS |
| Use case Id(s) | A02 & F06 |
| Requirement(s) | A02.FR.02, A02.FR.03, A02.FR.06, A02.FR.08, A02.FR.09 & F06.FR.04,F06.FR.05,F06.FR.10 |
| System under test | Charging Station |
| Description | The CSMS is able to request the Charging Station to update its charging station certificate using the TriggerMessageRequest message. |
| Purpose | To verify if the Charging Station is able to update its Charging Station Certificate. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The charging station supports security profile 3 - The active NetworkConnectionProfile uses security profile 3. |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>RenewChargingStationCertificate</i> for certificateType <i>ChargingStationCertificate</i> | |

| |
|--|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_A_12_CS: Update Charging Station Certificate by request of CSMS - Success - V2G Certificate

| | |
|--------------------------|--|
| Test case name | Update Charging Station Certificate by request of CSMS - Success - V2G Certificate |
| Test case Id | TC_A_12_CS |
| Use case Id(s) | A02 & F06 |
| Requirement(s) | A02.FR.02, A02.FR.03, A02.FR.06,A02.FR.13,A02.FR.15 & F06.FR.04,F06.FR.05,F06.FR.10 |
| System under test | Charging Station |
| Description | The CSMS is able to request the Charging Station to update its charging station certificate using the TriggerMessageRequest message. |
| Purpose | To verify if the Charging Station is able to update its V2G Charging Station Certificate. |
| Prerequisite(s) | The Charging Station supports ISO 15118. |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Memory State <i>RenewV2GChargingStationCertificate</i> | |

| |
|--|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_A_14_CS: Update Charging Station Certificate by request of CSMS - Invalid certificate

| | |
|-------------------|--|
| Test case name | Update Charging Station Certificate by request of CSMS - Invalid certificate |
| Test case Id | TC_A_14_CS |
| Use case Id(s) | A02 |
| Requirement(s) | A02.FR.07,A03.FR.07 |
| System under test | Charging Station |
| Description | The CSMS is able to request the Charging Station to update its charging station certificate using the TriggerMessageRequest message. |
| Purpose | To verify if the Charging Station is able to discard an invalid certificate and report a security event. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The Charging Station supports security profile 3 - The active NetworkConnectionProfile uses security profile 3. |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a TriggerMessageResponse | 1. The Test System sends a TriggerMessageRequest With requestedMessage <i>SignChargingStationCertificate</i> |
| 3 The Charging Station sends a SignCertificateRequest | 4. The Test System responds with a SignCertificateResponse With status <i>Accepted</i> |
| 6. The Charging Station responds with a CertificateSignedResponse | 5. The Test System sends a CertificateSignedRequest With certificateChain <i><Configured invalid_signingCertificate></i> certificateType <i>ChargingStationCertificate</i> |
| 7 The Charging Station sends a SecurityEventNotificationRequest | 8 The Test System responds with a SecurityEventNotificationResponse |

| |
|--|
| Tool validations |
| <p>* Step 2: Message: TriggerMessageResponse - status must be <i>Accepted</i></p> <p>* Step 3: Message: SignCertificateRequest - csr must contain <i><An CSR that meets the following requirements: When using RSA or DSA the key must be at least 2048 bits long. and when using elliptic curve cryptography the key must be at least 224 bits long. The received CSR must be transmitted as described in RFC 2986 and then encoded in Privacy-Enhanced Mail (PEM) format.></i></p> <p>* Step 6: Message: CertificateSignedResponse - status must be <i>Rejected</i></p> <p>* Step 7: Message: SecurityEventNotificationRequest - type must be <i>InvalidChargingStationCertificate</i></p> <p>Post scenario validations: N/a</p> |

TC_A_15_CS: Update Charging Station Certificate by request of CSMS - SignCertificateRequest Rejected

| | |
|-------------------|--|
| Test case name | Update Charging Station Certificate by request of CSMS - SignCertificateRequest Rejected |
| Test case Id | TC_A_15_CS |
| Use case Id(s) | A02 |
| Requirement(s) | N/a |
| System under test | Charging Station |
| Description | The CSMS is able to request the Charging Station to update its charging station certificate using the TriggerMessageRequest message. |
| Purpose | To verify if the Charging Station is able to discard an invalid certificate and report a security event. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The Charging Station supports security profile 3 - The active NetworkConnectionProfile uses security profile 3. |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a TriggerMessageResponse | 1. The Test System sends a TriggerMessageRequest With requestedMessage <i>SignChargingStationCertificate</i> |
| 3 The Charging Station sends a SignCertificateRequest | 4. The Test System responds with a SignCertificateResponse With status <i>Rejected</i> |

| |
|--|
| Tool validations |
| * Step 2: Message: TriggerMessageResponse - status must be <i>Accepted</i> |
| Post scenario validations: N/a |

TC_A_23_CS: Update Charging Station Certificate by request of CSMS - CertificateSignedRequest Timeout

| | |
|-------------------|---|
| Test case name | Update Charging Station Certificate by request of CSMS - CertificateSignedRequest Timeout |
| Test case Id | TC_A_23_CS |
| Use case Id(s) | A02 & F06 |
| Requirement(s) | A02.FR.17,A02.FR.18 |
| System under test | Charging Station |
| Description | The CSMS is able to request the Charging Station to update its charging station certificate using the TriggerMessageRequest message. |
| Purpose | To verify if the Charging Station is able to send a new signCertificateRequest when it did not receive a certificateSignedRequest after the configured timeout. CSMS will after a delay send a CertificateSignedRequest for each SignCertificateRequest that it has accepted. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The charging station supports security profile 3 - The Charging Station supports the CertificateSignedRequest Timeout feature |

| |
|---|
| Before (Preparations) |
| Configuration State: SecurityCtrlr.CertSigningWaitMinimum is <Configured CertSigningWaitMinimum> SecurityCtrlr.CertSigningRepeatTimes is 1 |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a TriggerMessageResponse | 1. The Test System sends a TriggerMessageRequest With requestedMessage <i>SignChargingStationCertificate</i> |
| 3 The Charging Station sends a SignCertificateRequest | 4. The Test System responds with a SignCertificateResponse With status <i>Accepted</i> |
| 6 The Charging Station sends a SignCertificateRequest | 5. The Test System does NOT send the CertificateSignedRequest and waits for the SignCertificateRequest to be resend after the <Configured CertSigningWaitMinimum> |
| | 7. The Test System responds with a SignCertificateResponse With status <i>Accepted</i> |
| 9 The Charging Station sends a SignCertificateRequest | 8. The Test System does NOT send the CertificateSignedRequest and waits for the SignCertificateRequest to be resend after the <Configured CertSigningWaitMinimum> times 2 |
| | 10. The Test System responds with a SignCertificateResponse With status <i>Accepted</i> |
| 12. The Charging Station responds with a CertificateSignedResponse | 11. The Test System sends a CertificateSignedRequest With certificateChain <Certificate generated from the received CSR from step 3 and signed by the provided CSMS Root certificate> certificateType <i>ChargingStationCertificate</i> |

| Main (Test scenario) | |
|---|--|
| 14. The Charging Station responds with a CertificateSignedResponse | 13. The Test System sends a CertificateSignedRequest With certificateChain <Certificate generated from the received CSR from step 6 and signed by the provided CSMS Root certificate> certificateType ChargingStationCertificate |
| 16. The Charging Station responds with a CertificateSignedResponse | 15. The Test System sends a CertificateSignedRequest With certificateChain <Certificate generated from the received CSR from step 9 and signed by the provided CSMS Root certificate> certificateType ChargingStationCertificate |

| Tool validations |
|---|
| <p>* Step 2: Message: TriggerMessageResponse - status must be Accepted</p> <p>* Step 3/6/9: Message: SignCertificateRequest - csr must contain <An CSR that meets the following requirements: When using RSA or DSA the key must be at least 2048 bits long. and when using elliptic curve cryptography the key must be at least 224 bits long. The received CSR must be transmitted as described in RFC 2986 and then encoded in Privacy-Enhanced Mail (PEM) format.></p> <p>* Step 5: - The Charging Station shall not resend the SignCertificateRequest before the <Configured CertSigningWaitMinimum> expired</p> <p>* Step 8: - The Charging Station shall not resend the SignCertificateRequest before the <Configured CertSigningWaitMinimum> times 2 expired</p> <p>* Step 12, 14, 16: Message: CertificateSignedResponse - status must be Accepted or Rejected</p> |
| <p>Post scenario validations: Note: It does not matter whether Charging Station accepts first or last or all certificates. At least one CertificateSignedResponse must have status Accepted</p> |

TC_A_19_CS: Upgrade Charging Station Security Profile - Accepted

| | |
|-------------------|--|
| Test case name | Upgrade Charging Station Security Profile - Accepted |
| Test case Id | TC_A_19_CS |
| Use case Id(s) | A05 |
| Requirement(s) | A05.FR.04,A05.FR.05,A05.FR.06 |
| System under test | Charging Station |
| Description | The CSMS updates the connection details on the Charging Station, to increase the security profile level. |
| Purpose | To verify if the Charging Station is able to increase the security profile level when configured to do so by the CSMS. |
| Prerequisite(s) | Security profile must be set to 1 or 2 |

Before (Preparations)

Configuration State:

N/a

Memory State:

If configured <Security profile> is 1, then [CertificateInstalled](#)If configured <Security profile> is 2, then [RenewChargingStationCertificate](#)

Reusable State(s):

N/a

Main (Test scenario)

| Charging Station | CSMS |
|--|--|
| 2. The Charging Station responds with a SetNetworkProfileResponse | <p>1. The Test System sends a SetNetworkProfileRequest with configurationSlot is <Configured configurationSlot> or <Configured configurationSlot2> depending on which one is already in use</p> <ul style="list-style-type: none"> - connectionData.messageTimeout <Configured messageTimeout> - connectionData.ocppCsmsUrl <Configured ocppCsmsUrl> - connectionData.ocppInterface <Configured ocppInterface> - connectionData.ocppVersion OCPP20 - connectionData.securityProfile <Configured securityProfile + 1> <p><u>Note(s):</u></p> <ul style="list-style-type: none"> - The Test System checks the ActiveNetworkProfile variable to see which slot is currently active. - The Test System prevents overwriting the NetworkProfile at the active slot, as this is not recommended. |
| 4. The Charging Station responds with a SetVariablesResponse | <p>3. The Test System sends a SetVariablesRequest with variable.name is "NetworkConfigurationPriority"</p> <p>component.name is "OCPPCommCtrlr"</p> <p>attributeValue is <configurationSlot set at Step 1, the other configured configurationSlot></p> |
| 6. The Charging Station responds with a ResetResponse | <p>5. The Test System sends a ResetRequest with type OnIdle</p> |
| | <p>7. The Test System restarts the WebSocket server using <Configured securityProfile + 1></p> |
| 8. The Charging Station reconnects to the Test System using <Configured securityProfile + 1> | <p>9. The Test System accepts the connection attempt.</p> |
| 10. Execute Reusable State Booted | |
| 12. The Charging Station responds with GetVariablesResponse | <p>11. Test System sends GetVariablesRequest with:</p> <ul style="list-style-type: none"> - variable.name = "SecurityProfile" - component.name = "SecurityCtrlr" |

| Main (Test scenario) | |
|---|---|
| 14. The Charging Station responds with GetVariablesResponse | 13. Test System sends GetVariablesRequest with: - variable.name = "NetworkConfigurationPriority" - component.name = "OCPPCommCtrlr" |
| <i>The following steps are only executed when this testcase is upgrading from Security Profile 1 to Security Profile 2.</i> | |
| 16. The Charging Station does NOT reconnect to the Test System using Security Profile 1. | 15. The Test System closes the connection and restarts the WebSocket server using Security profile 1 and waits the <Configured long operation timeout>. |
| 18. The Charging Station reconnects to the Test System using Security Profile 2. | 17. The Test System restarts the WebSocket server using Security Profile 2. |

| Tool validations |
|--|
| <p>* Step 2: Message SetNetworkProfileResponse - status <i>Accepted</i></p> <p>* Step 4: Message SetVariablesResponse - setVariableResult[0].attributeStatus <i>Accepted</i> OR <i>RebootRequired</i></p> <p>* Step 6: Message ResetResponse - status <i>Accepted</i></p> <p>* Step 12: Message GetVariablesResponse - getVariableResult[0].attributeValue <i><Configured securityProfile + 1></i></p> <p>* Step 14: Message GetVariablesResponse - getVariableResult[0].attributeValue Does not contain the configurationSlot with the previous (lower) security profile</p> <p>Post scenario validations: - N/a</p> |

TC_A_20_CS: Upgrade Charging Station Security Profile - No valid CSMSRootCertificate installed

| | |
|-------------------|---|
| Test case name | Upgrade Charging Station Security Profile - No valid CSMSRootCertificate installed |
| Test case Id | TC_A_20_CS |
| Use case Id(s) | A05 |
| Requirement(s) | A05.FR.02 |
| System under test | Charging Station |
| Description | The CSMS is able to change the connectionData at the Charging Station. By doing this it is able to upgrade the connection to a higher security profile. |
| Purpose | To verify if the Charging Station is able to reject upgrading to a higher security profile when it does not have a valid CSMSRootCertificate installed. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The Test System connectionData configuration for SUT Charging Station only allows for ip addresses the Test System is able to bind. - The Charging Station supports at least 2 security profiles, one of which is security profile 1. - The Charging Station does not have a valid CSMSRootCertificate installed. - The first Test System connectionData configuration slot must be configured for security profile 1. - The second Test System connectionData configuration slot must be configured for security profile 2 or 3. - The Charging Station is connected using security profile 1. - When starting this testcase the Test System will start another webSocket server for the second connectionData slot. |

Before (Preparations)

Configuration State:
OCPPCommCtrlr.NetworkConfigurationPriority is <ActiveNetworkProfile slot> (All others are removed)

Memory State:
N/a

Reusable State(s):
N/a

Main (Test scenario)

| Charging Station | CSMS |
|--|---|
| 2. The Charging Station responds with a SetNetworkProfileResponse | <p>1. The Test System sends a SetNetworkProfileRequest with - configurationSlot is <Configured configurationSlot2> or <Configured configurationSlot> (the one currently not used for the active connection)</p> <ul style="list-style-type: none"> - connectionData.messageTimeout <Configured messageTimeout2> - connectionData.ocppCsmsUrl <ocppCsmsUrl that is not currently active> - connectionData.ocppInterface <Configured ocppInterface2> - connectionData.ocppVersion OCPP20 - connectionData.securityProfile <Configured securityProfile2> |
| 4. The Charging Station responds with a SetVariablesResponse | <p>3. The Test System sends a SetVariablesRequest with variable.name is "NetworkConfigurationPriority"</p> <p>component.name is "OCPPCommCtrlr"</p> <p>attributeValue is <configurationSlot set at step 1>,<previous configurationSlot></p> |

| Tool validations |
|---|
| <p>* Step 2:</p> <p>Message SetNetworkProfileResponse</p> <p>- status <i>Accepted</i> or <i>Rejected</i></p> <p>* Step 4:</p> <p>Message SetVariablesResponse</p> <p>- setVariableResult[0].attributeStatus <i>Rejected</i></p> |
| <p>Post scenario validations:</p> <p>- N/a</p> |

TC_A_21_CS: Upgrade Charging Station Security Profile - No valid ChargingStationCertificate installed

| | |
|-------------------|---|
| Test case name | Upgrade Charging Station Security Profile - No valid ChargingStationCertificate installed |
| Test case Id | TC_A_21_CS |
| Use case Id(s) | A05 |
| Requirement(s) | A05.FR.03 |
| System under test | Charging Station |
| Description | The CSMS is able to change the connectionData at the Charging Station. By doing this it is able to upgrade the connection to a security profile 3. |
| Purpose | To verify if the Charging Station is able to reject upgrading to a security profile 3 when it does not have a valid ChargingStationCertificate installed. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The Test System connectionData configuration for SUT Charging Station only allows for ip addresses the Test System is able to bind. - The Charging Station support at least 2 security profiles. - The Charging Station does not have a valid ChargingStationCertificate installed. - The Charging Station has a valid CSMSRootCertificate installed. - When starting this testcase the Test System will start another webSocket server for the second connectionData slot. |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a SetNetworkProfileResponse | 1. The Test System sends a SetNetworkProfileRequest with - configurationSlot is <Configured configurationSlot> or <Configured configurationSlot> depending on which one is already in use - connectionData.messageTimeout <Configured messageTimeout2> - connectionData.ocppCsmsUrl <ocppCsmsUrl that is not currently active> - connectionData.ocppInterface <Configured ocppInterface2> - connectionData.ocppVersion OCPP20 - connectionData.securityProfile 3 |
| 4. The Charging Station responds with a SetVariablesResponse | 3. The Test System sends a SetVariablesRequest with variable.name is "NetworkConfigurationPriority" component.name is "OCPPCommCtrlr" attributeValue is <Configured configurationSlot2>,<Configured configurationSlot> |

| |
|---|
| Tool validations |
| * Step 2: Message SetNetworkProfileResponse - status Accepted * Step 4: Message SetVariablesResponse - setVariableResult[0].attributeStatus Rejected |

| |
|--|
| Tool validations |
| Post scenario validations: - N/a |

TC_A_22_CS: Upgrade Charging Station Security Profile - Downgrade security profile - Rejected

| | |
|-------------------|--|
| Test case name | Upgrade Charging Station Security Profile - Downgrade security profile - Rejected |
| Test case Id | TC_A_22_CS |
| Use case Id(s) | A05, B09 |
| Requirement(s) | B09.FR.04 |
| System under test | Charging Station |
| Description | The CSMS is able to change the connectionData at the Charging Station. It tries to downgrade the connection to security profile 1. |
| Purpose | To verify if the Charging Station is able to reject downgrading to security profile 1. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The Test System connectionData configuration for SUT Charging Station only allows for ip addresses the Test System is able to bind. - The Charging Station supports security profile 2 and/or 3. - The Charging Station has a connection using security profile 2 or 3. - When starting this testcase the Test System will start another websocket server for the second connectionData slot. |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|--|---|
| Charging Station | CSMS |
| 2. The Charging Station responds with a SetNetworkProfileResponse | <p>1. The Test System sends a SetNetworkProfileRequest with:</p> <ul style="list-style-type: none">- configurationSlot is <i><Configured configurationSlot></i> or <i><Configured configurationSlot2></i> depending on which one is already in use- connectionData.messageTimeout <i><Configured messageTimeout2></i>- connectionData.ocppCsmsUrl <i><ocppCsmsUrl that is not currently active></i>- connectionData.ocppInterface <i><Configured ocppInterface2></i>- connectionData.ocppVersion <i>OCPP20</i>- connectionData.securityProfile <i>1</i> |

| |
|--|
| Tool validations |
| * Step 2: Message SetNetworkProfileResponse - status <i>Rejected</i> |
| Post scenario validations: - N/a |

B Provisioning**TC_B_01_CS: Cold Boot Charging Station - Accepted**

| | |
|--------------------------|---|
| Test case name | Cold Boot Charging Station - Accepted |
| Test case Id | TC_B_01_CS |
| Use case Id(s) | B01 |
| Requirement(s) | B01.FR.01, B01.FR.05, B01.FR.09 |
| System under test | Charging Station |
| Description | The booting mechanism allows a Charging Station to provide some general information about the Charging Station to the CSMS on startup AND it allows the Charging Station to request whether it is allowed to start sending other OCPP messages. |
| Purpose | To verify whether the Charging Station is able to perform the booting mechanism as described at the OCPP specification. |
| Prerequisite(s) | N/a |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>Booted</i> | |

| |
|---|
| Tool validations |
| N/a |
| Post scenario validations: - A message to report the state of a connector has been received for all connectors. |

TC_B_02_CS: Cold Boot Charging Station - Pending

| | |
|-------------------|---|
| Test case name | Cold Boot Charging Station - Pending |
| Test case Id | TC_B_02_CS |
| Use case Id(s) | B02, F06 |
| Requirement(s) | B02.FR.01, B02.FR.02, B02.FR.04, B02.FR.05, B02.FR.06, B02.FR.08, F06.FR.17 |
| System under test | Charging Station |
| Description | The booting mechanism allows a Charging Station to provide some general information about the Charging Station to the CSMS on startup AND it allows the Charging Station to request whether it is allowed to start sending other OCPP messages. A CSMS can temporarily halt the Charging Stations operations by returning the Pending status at the BootNotificationResponse. During this time the CSMS is able to retrieve and set configurations from the Charging Station. |
| Purpose | To verify whether the Charging Station is able to correctly handle the pending state of the boot mechanism. |
| Prerequisite(s) | The testcases; TC_B_06_CS, TC_B_09_CS, TC_B_13_CS are executed with test result PASS. |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action</u> : Reboot the Charging Station. | |
| 1. The Charging Station sends a BootNotificationRequest | 2. The Test System responds with a BootNotificationResponse with status <i>Pending</i> interval <i><Configured heartbeatInterval></i> |
| 4. The Charging Station responds with SetVariablesResponse | 3. Test System sends SetVariablesRequest with: - variable.name = "OfflineThreshold" - component.name = "OCPPCommCtrlr" - attributeValue = "300" - attributeType is omitted |
| 6. The Charging Station responds with GetVariablesResponse | 5. Test System sends GetVariablesRequest with: - variable.name = "OfflineThreshold" - component.name = "OCPPCommCtrlr" - attributeType is omitted |
| 8. Charging Station responds with: GetBaseReportResponse | 7. Test System sends GetBaseReportRequest with: - requestId = <i><Generated requestId></i> - reportBase = <i>FullInventory</i> |
| Charging Station | CSMS |
| 9. Charging Station responds with: NotifyReportRequest | 10. Test System sends NotifyReportResponse |
| <u>Note(s)</u> : - This step is repeated as often as needed to report all configuration variables. | |

| Main (Test scenario) | |
|---|--|
| 12. The Charging Station responds with a RequestStartTransactionResponse | 11. The Test System sends a RequestStartTransactionRequest <u>Note(s):</u> - This step is executed after the Test System received all NotifyReport messages. This is indicated by the tbc and seqNo fields. |
| 14. The Charging Station responds with a TriggerMessageResponse | 13. The Test System sends a TriggerMessageRequest with requestedMessage <i>BootNotification</i> |
| 15. The Charging Station sends a BootNotificationRequest <u>Note(s):</u> - The Charging Station resends the BootNotificationRequest after having responded to the TriggerMessageRequest, so before the interval from the BootNotificationResponse has been passed. | 16. The Test System responds with a BootNotificationResponse with status <i>Accepted</i> and interval <i><Configured heartbeatInterval></i> |
| 17. The Charging Station notifies the CSMS about the current state of all connectors. | 18. The Test System responds accordingly. |

| Tool validations |
|---|
| <p>* Step 4: Message: SetVariablesResponse - setVariableResult[0].attributeStatus <i>Accepted</i></p> <p>* Step 6: Message: GetVariablesResponse - getVariableResult[0].attributeStatus <i>Accepted</i></p> <p>* Step 8: Message: GetBaseReportResponse - status <i>Accepted</i></p> <p>* Step 12: Message: RequestStartTransactionResponse - status <i>Rejected</i></p> <p>* Step 14: Message: TriggerMessageResponse - status <i>Accepted</i> or <i>NotImplemented</i></p> <p>* Step 15: Message: BootNotificationRequest - reason <i>Triggered</i> (If the status from the response from step 14 contained <i>Accepted</i>)</p> <p>* Step 17: Message: StatusNotificationRequest - connectorStatus <i>Available</i></p> <p>Message: NotifyEventRequest - eventData[0].trigger <i>Delta</i> - eventData[0].actualValue <i>"Available"</i> - eventData[0].component.name <i>"Connector"</i> - eventData[0].variable.name <i>"AvailabilityState"</i></p> <p>Post scenario validations: - A message to report the state of a connector has been received for all connectors.</p> |

TC_B_03_CS: Cold Boot Charging Station - Rejected

| | |
|-------------------|---|
| Test case name | Cold Boot Charging Station - Rejected |
| Test case Id | TC_B_03_CS |
| Use case Id(s) | B03 |
| Requirement(s) | B03.FR.02, B03.FR.04, B03.FR.06 |
| System under test | Charging Station |
| Description | The booting mechanism allows a Charging Station to provide some general information about the Charging Station to the CSMS on startup AND it allows the Charging Station to request whether it is allowed to start sending other OCPP messages. |
| Purpose | To verify whether the Charging Station is able to correctly handle a rejected BootNotification. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| Manual Action: Reboot the Charging Station. | |
| 1. The Charging Station sends a BootNotificationRequest | 2. The Test System responds with a BootNotificationResponse with status <i>Rejected</i> interval <Configured heartbeatInterval> |
| 3. The Charging Station sends a BootNotificationRequest <u>Note(s):</u> - The Charging Station resends the <i>BootNotificationRequest</i> after <i>x</i> seconds, whereby <i>x</i> is equal to or greater than the interval from the <i>BootNotificationResponse</i> . - The Charging Station is not allowed to send any OCPP message in the meantime. - The Charging Station is allowed to close the connection until it needs to resend the <i>BootNotificationRequest</i> . | 4. The Test System responds with a BootNotificationResponse with status <i>Accepted</i> interval <Configured heartbeatInterval> |
| 5. The Charging Station notifies the CSMS about the current state of all connectors. | 6. The Test System responds accordingly. |

| |
|---|
| Tool validations |
| <p>* Step 5:</p> <p>Message: StatusNotificationRequest</p> <ul style="list-style-type: none"> - connectorStatus <i>Available</i> <p>Message: NotifyEventRequest</p> <ul style="list-style-type: none"> - eventData[0].trigger <i>Delta</i> - eventData[0].actualValue <i>"Available"</i> - eventData[0].component.name <i>"Connector"</i> - eventData[0].variable.name <i>"AvailabilityState"</i> |
| <p>Post scenario validations:</p> <ul style="list-style-type: none"> - A message to report the state of a connector has been received for all connectors. |

TC_B_30_CS: Cold Boot Charging Station - Pending/Rejected - SecurityError

| | |
|-------------------|---|
| Test case name | Cold Boot Charging Station - Pending/Rejected - SecurityError |
| Test case Id | TC_B_30_CS |
| Use case Id(s) | B03 |
| Requirement(s) | B03.FR.08 |
| System under test | Charging Station |
| Description | The booting mechanism allows a Charging Station to provide some general information about the Charging Station to the CSMS on startup AND it allows the Charging Station to request whether it is allowed to start sending other OCPP messages. The CSMS may respond to the BootNotificationRequest with status <i>Rejected</i> . During this state, the Charging Station is not allowed to send RPC Framework: CALL message that is NOT a BootNotificationRequest. |
| Purpose | To verify whether the Charging Station is able to handle unauthorized messages from the CSMS by responding with a SecurityError. |
| Prerequisite(s) | |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State(s):

N/a

Main (Test scenario)

| Charging Station | CSMS |
|--|--|
| 1. The Charging Station sends a BootNotificationRequest | 2. The Test System responds with a BootNotificationResponse with status <i>Rejected</i> |
| 4. The Charging Station responds with RPC Framework: CALLERROR: SecurityError. | 3. The Test System sends a GetBaseReportRequest with reportBase <i>FullInventory</i> <u>Note(s)</u> : The Test System will only send this request if the Charging Station does not disconnect |

Tool validations

N/a

N/a

TC_B_06_CS: Get Variables - single value

| | |
|-------------------|---|
| Test case name | Get Variables - single value |
| Test case Id | TC_B_06_CS |
| Use case Id(s) | B06 |
| Requirement(s) | B06.FR.01, B06.FR.02, B06.FR.03, B06.FR.04, B06.FR.10, B06.FR.11 |
| System under test | Charging Station |
| Description | Get the value of one of the required variables of OCPPCommCtrlr |
| Purpose | To test getting a single value using GetVariablesRequest for one of the mandatory component/variable combinations that must exist in the DM implementation. |
| Prerequisite(s) | N/A |

| |
|---|
| Before (Preparations) |
| Configuration State: OCPPCommCtrlr.OfflineThreshold is 300 |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. Charging Station responds with GetVariablesResponse | 1. Test System sends GetVariablesRequest with: - variable.name = "OfflineThreshold" - component.name = "OCPPCommCtrlr" - attributeType = Actual |

| |
|---|
| Tool validations |
| * Step 2: Message: GetVariablesResponse - attributeStatus = Accepted - attributeType = Actual - attributeValue = "300" - component.name = "OCPPCommCtrlr" - variable.name = "OfflineThreshold" - attributeStatusInfo is absent or attributeStatusInfo.reasonCode = "NoError" |
| Post scenario validations: N/A |

TC_B_07_CS: Get Variables - multiple values

| | |
|-------------------|--|
| Test case name | Get Variables - multiple values |
| Test case Id | TC_B_07_CS |
| Use case Id(s) | B06 |
| Requirement(s) | B06.FR.01, B06.FR.02, B06.FR.03, B06.FR.04, B06.FR.10 |
| System under test | Charging Station |
| Description | Get the value of two required variables |
| Purpose | To test getting multiple values using GetVariablesRequest for one of the mandatory component/variable combinations that must exist in the DM implementation. |
| Prerequisite(s) | N/A |

| |
|---|
| Before (Preparations) |
| Configuration State: OCPPCommCtrlr.OfflineThreshold is 300 AuthCtrlr.LocalAuthorizeOffline is true |
| Memory State: N/a |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|---|--|
| Charging Station | CSMS |
| 2. Charging Station responds with GetVariablesResponse with attributeStatus = <i>Accepted</i> . | 1. Test System sends GetVariablesRequest with: - getVariableData[0].variable.name = <i>"OfflineThreshold"</i> - getVariableData[0].component.name = <i>"OCPPCommCtrlr"</i> - getVariableData[0].attributeType = <i>Actual</i> - getVariableData[1].variable.name = <i>"LocalAuthorizeOffline"</i> - getVariableData[1].component.name = <i>"AuthCtrlr"</i> - getVariableData[1].attributeType = <i>Actual</i> |

| |
|---|
| Tool validations |
| <p>* Step 2:</p> <p>Message: GetVariablesResponse has (in arbitrary order)</p> <p>GetVariableResultType[0]:</p> <ul style="list-style-type: none"> - attributeStatus = Accepted - attributeType = Actual - attributeValue = 300 - component.name = "OCPPCommCtrlr" - variable.name = "OfflineThreshold" - attributeStatusInfo is absent or attributeStatusInfo.reasonCode = "NoError" <p>GetVariableResultType[1]:</p> <ul style="list-style-type: none"> - attributeStatus = Accepted - attributeType = Actual - attributeValue = "true" - component.name = "AuthCtrlr" - variable.name = "LocalAuthorizeOffline" - attributeStatusInfo is absent or attributeStatusInfo.reasonCode = "NoError" <p>Post scenario validations:</p> <p>N/A</p> |

TC_B_32_CS: Get Variables - Unknown component

| | |
|-------------------|--|
| Test case name | Get Variables - Unknown component |
| Test case Id | TC_B_32_CS |
| Use case Id(s) | B06 |
| Requirement(s) | B06.FR.06 |
| System under test | Charging Station |
| Description | The CSMS can use a GetVariablesRequest to retrieve values from device model variables at the Charging Station. |
| Purpose | To verify whether the Charging Station can handle receiving a GetVariablesRequest for an unknown component. |
| Prerequisite(s) | N/A |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with GetVariablesResponse | 1. Test System sends GetVariablesRequest with: - variable.name = "OfflineThreshold" - component.name = "UnknownComponent" - attributeType is omitted |

| |
|---|
| Tool validations |
| * Step 2: Message: GetVariablesResponse - getVariableResult[0].attributeStatus = <i>UnknownComponent</i> - getVariableResult[0].component.name = "UnknownComponent" - getVariableResult[0].variable.name = "OfflineThreshold" |
| Post scenario validations: N/A |

TC_B_33_CS: Get Variables - Unknown variable

| | |
|-------------------|--|
| Test case name | Get Variables - Unknown variable |
| Test case Id | TC_B_33_CS |
| Use case Id(s) | B06 |
| Requirement(s) | B06.FR.07 |
| System under test | Charging Station |
| Description | The CSMS can use a GetVariablesRequest to retrieve values from device model variables at the Charging Station. |
| Purpose | To verify whether the Charging Station can handle receiving a GetVariablesRequest for an unknown variable. |
| Prerequisite(s) | N/A |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with GetVariablesResponse | 1. Test System sends GetVariablesRequest with: - variable.name = "UnknownVariable" - component.name = "OCPPCommCtrlr" - attributeType is omitted |

| |
|---|
| Tool validations |
| * Step 2: Message: GetVariablesResponse - getVariableResult[0].attributeStatus = UnknownVariable - getVariableResult[0].component.name = "OCPPCommCtrlr" - getVariableResult[0].variable.name = "UnknownVariable" |
| Post scenario validations: N/A |

TC_B_34_CS: Get Variables - Not supported attribute type

| | |
|-------------------|---|
| Test case name | Get Variables - Not supported attribute type |
| Test case Id | TC_B_34_CS |
| Use case Id(s) | B06 |
| Requirement(s) | B06.FR.08 |
| System under test | Charging Station |
| Description | The CSMS can use a GetVariablesRequest to retrieve values from device model variables at the Charging Station. |
| Purpose | To verify whether the Charging Station can handle receiving a GetVariablesRequest for a not supported attribute type. |
| Prerequisite(s) | N/A |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with GetVariablesResponse | 1. Test System sends GetVariablesRequest with: - variable.name = "OfflineThreshold" - component.name = "OCPPCommCtrlr" - attributeType = Target |

| |
|---|
| Tool validations |
| * Step 2: Message: GetVariablesResponse - getVariableResult[0].attributeStatus = <i>NotSupportedAttributeType</i> - getVariableResult[0].component.name = "OCPPCommCtrlr" - getVariableResult[0].variable.name = "OfflineThreshold" - getVariableResult[0].attributeType = Target |
| Post scenario validations: N/A |

TC_B_09_CS: Set Variables - single value

| | |
|-------------------|---|
| Test case name | Set Variables - single value |
| Test case Id | TC_B_09_CS |
| Use case Id(s) | B05 |
| Requirement(s) | B05.FR.01, B05.FR.02, B05.FR.03, B05.FR.10, B05.FR.12 |
| System under test | Charging Station |
| Description | Set the value of one of the required variables of OCPPCommCtrlr |
| Purpose | To test setting a single value using SetVariablesRequest for one of the mandatory component/variable combinations that must exist in the DM implementation. |
| Prerequisite(s) | N/A |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|---|---|
| Charging Station | CSMS |
| 2. Charging Station responds with SetVariablesResponse with attributeStatus = <i>Accepted</i> . | 1. Test System sends SetVariablesRequest with: - variable.name = <i>"OfflineThreshold"</i> - component.name = <i>"OCPPCommCtrlr"</i> - attributeValue = <i>"300"</i> - attributeType <i>Actual</i> |

| |
|---|
| Tool validations |
| * Step 2: Message: SetVariablesResponse - setVariableResult[0].attributeStatus = <i>Accepted</i> - setVariableResult[0].attributeType = <i>Actual</i> - setVariableResult[0].component.name = <i>"OCPPCommCtrlr"</i> - setVariableResult[0].variable.name = <i>"OfflineThreshold"</i> - setVariableResult[0].attributeStatusInfo is absent or setVariableResult[0].attributeStatusInfo.reasonCode = <i>"NoError"</i> |
| Post scenario validations: N/A |

TC_B_10_CS: Set Variables - multiple values

| | |
|-------------------|--|
| Test case name | Set Variables - multiple values |
| Test case Id | TC_B_10_CS |
| Use case Id(s) | B05 |
| Requirement(s) | B05.FR.01, B05.FR.02, B05.FR.03, B05.FR.10, B05.FR.12 |
| System under test | Charging Station |
| Description | Set the value of two required variables |
| Purpose | To test setting multiple values using GetVariablesRequest for one of the mandatory component/variable combinations that must exist in the DM implementation. |
| Prerequisite(s) | N/A |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|---|--|
| Charging Station | CSMS |
| 2. Charging Station responds with SetVariablesResponse with attributeStatus = <i>Accepted</i> . | <p>1. Test System sends SetVariablesRequest with:</p> <ul style="list-style-type: none">- setVariableData[0].variable.name = "OfflineThreshold"- setVariableData[0].component.name = "OCPPCommCtrlr"- setVariableData[0].attributeValue = "300"- setVariableData[0].attributeType = Actual -setVariableData[1].variable.name = "LocalAuthorizeOffline"- setVariableData[1].component.name = "AuthCtrlr"- setVariableData[1].attributeValue = "true"- setVariableData[0].attributeType = Actual |

| |
|--|
| Tool validations |
| <p>* Step 2:</p> <p>Message: SetVariablesResponse has (in arbitrary order)</p> <p>SetVariableResultType[1]:</p> <ul style="list-style-type: none"> - attributeStatus = Accepted - attributeType = Actual - component.name = "OCPPCommCtrlr" - variable.name = "OfflineThreshold" - attributeStatusInfo is absent or attributeStatusInfo.reasonCode = "NoError" <p>SetVariableResultType[2]:</p> <ul style="list-style-type: none"> - attributeStatus = Accepted - attributeType = Actual - component.name = "AuthCtrlr" - variable.name = "LocalAuthorizeOffline" - attributeStatusInfo is absent or attributeStatusInfo.reasonCode = "NoError" <p>Post scenario validations:</p> <p>N/A</p> |

TC_B_35_CS: Set Variables - Unknown component

| | |
|-------------------|---|
| Test case name | Set Variables - Unknown component |
| Test case Id | TC_B_35_CS |
| Use case Id(s) | B05 |
| Requirement(s) | B05.FR.04 |
| System under test | Charging Station |
| Description | The CSMS can use a SetVariablesRequest to set values from device model variables at the Charging Station. |
| Purpose | To verify whether the Charging Station can handle receiving a SetVariablesRequest for an unknown component. |
| Prerequisite(s) | N/A |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with SetVariablesResponse | 1. Test System sends SetVariablesRequest with: - variable.name = "OfflineThreshold" - component.name = "UnknownComponent" - attributeType is omitted |

| |
|---|
| Tool validations |
| * Step 2: Message: SetVariablesResponse - setVariableResult[0].attributeStatus = <i>UnknownComponent</i> - setVariableResult[0].component.name = "UnknownComponent" - setVariableResult[0].variable.name = "OfflineThreshold" |
| Post scenario validations: N/A |

TC_B_36_CS: Set Variables - Unknown variable

| | |
|-------------------|--|
| Test case name | Set Variables - Unknown variable |
| Test case Id | TC_B_36_CS |
| Use case Id(s) | B05 |
| Requirement(s) | B05.FR.05 |
| System under test | Charging Station |
| Description | The CSMS can use a SetVariablesRequest to set values from device model variables at the Charging Station. |
| Purpose | To verify whether the Charging Station can handle receiving a SetVariablesRequest for an unknown variable. |
| Prerequisite(s) | N/A |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|---|---|
| Charging Station | CSMS |
| 2. The Charging Station responds with SetVariablesResponse | 1. Test System sends SetVariablesRequest with: - variable.name = "UnknownVariable" - component.name = "OCPPCommCtrlr" - attributeType is omitted |

| |
|---|
| Tool validations |
| * Step 2: Message: SetVariablesResponse - setVariableResult[0].attributeStatus = UnknownVariable - setVariableResult[0].component.name = "OCPPCommCtrlr" - setVariableResult[0].variable.name = "UnknownVariable" |
| Post scenario validations: N/A |

TC_B_37_CS: Set Variables - Not supported attribute type

| | |
|-------------------|---|
| Test case name | Set Variables - Not supported attribute type |
| Test case Id | TC_B_37_CS |
| Use case Id(s) | B05 |
| Requirement(s) | B05.FR.06 |
| System under test | Charging Station |
| Description | The CSMS can use a SetVariablesRequest to set values from device model variables at the Charging Station. |
| Purpose | To verify whether the Charging Station can handle receiving a SetVariablesRequest for a not supported attribute type. |
| Prerequisite(s) | N/A |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with SetVariablesResponse | 1. Test System sends SetVariablesRequest with: - variable.name = "OfflineThreshold" - component.name = "OCPPCommCtrlr" - attributeType = Target |

| |
|--|
| Tool validations |
| * Step 2: Message: SetVariablesResponse - setVariableResult[0].attributeStatus = NotSupportedAttributeType - setVariableResult[0].component.name = "OCPPCommCtrlr" - setVariableResult[0].variable.name = "OfflineThreshold" - setVariableResult[0].attributeType = Target |
| Post scenario validations: N/A |

TC_B_11_CS: Set Variables - invalidly formatted values

| | |
|-------------------|---|
| Test case name | Set Variables - invalidly formatted values |
| Test case Id | TC_B_11_CS |
| Use case Id(s) | B05 |
| Requirement(s) | B05.FR.07 |
| System under test | Charging Station |
| Description | Set the value of two of the required variables of OCPPCommCtrlr |
| Purpose | To test setting of variables of different type with invalidly formatted values using GetVariablesRequest for one of the mandatory component/variable combinations that must exist in the DM implementation. |
| Prerequisite(s) | Charging Station DM has the variable "NextTimeOffsetTransitionDateTime" of component "ClockCtrlr" to test setting of a date. |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Notes:</u> Steps 1 to 8 are repeated 5 times for value = <configured offlineThreshold>, <configured offlineThreshold + 0.1>, true, currentTime, "abc" | |
| 2. Charging Station responds with SetVariablesResponse with If value not supported: attributeStatus = Rejected attributeStatusInfo = InvalidValue If component/variable/value supported: attributeStatus = Accepted | 1. Test System sends SetVariablesRequest with - variable.name = "OfflineThreshold" - component.name = "OCPPCommCtrlr" - attributeValue = value |
| <u>Notes:</u> Steps 3 and 4 will only be tested if this component/variable combination is supported | |
| 4. Charging Station responds with SetVariablesResponse with If value not supported: attributeStatus = Rejected attributeStatusInfo = InvalidValue If component/variable/value supported: attributeStatus = Accepted | 3. Test System sends SetVariablesRequest with - variable.name = "LimitChangeSignificance" - component.name = "SmartChargingCtrlr" - attributeValue = value |
| <u>Notes:</u> Steps 5 and 6 will only be executed if this component/variable combination is readwrite | |
| 6. Charging Station responds with SetVariablesResponse with If value not supported: attributeStatus = Rejected attributeStatusInfo = InvalidValue If component/variable/value supported: attributeStatus = Accepted | 5. Test System sends SetVariablesRequest with: - variable.name = "AuthorizeRemoteStart" - component.name = "AuthCtrlr" - attributeValue = value |
| <u>Notes:</u> Steps 7 and 8 will only be executed if the CS supports this component/variable combination | |

| Main (Test scenario) | |
|---|---|
| <p>8. Charging Station responds with SetVariablesResponse with</p> <p><i>If value not supported:</i></p> <p>attributeStatus = <i>Rejected</i></p> <p>attributeStatusInfo = <i>InvalidValue</i></p> <p><i>If component/variable/value supported:</i></p> <p>attributeStatus = <i>Accepted</i></p> | <p>7. Test System sends SetVariablesRequest with:</p> <ul style="list-style-type: none"> - variable.name = <i>"NextTimeOffsetTransitionDateTime"</i> - component.name = <i>"ClockCtrlr"</i> - attributeValue = <i>value</i> |
| Tool validations | |
| <p>* Step 2:</p> <p>Message: SetVariablesResponse has</p> <p>SetVariableResultType</p> <ul style="list-style-type: none"> - attributeStatus = <i>Rejected/Accepted</i> - attributeType = <i>Actual</i> - component.name = <i>"OCPPCommCtrlr"</i> - variable.name = <i>"OfflineThreshold"</i> - attributeStatusInfo is absent or attributeStatusInfo.reasonCode = <i>InvalidValue</i> (not required) | |
| <p>* Step 4:</p> <p>Message: SetVariablesResponse has</p> <p>SetVariableResultType</p> <ul style="list-style-type: none"> - attributeStatus = <i>Rejected/Accepted</i> - attributeType = <i>Actual</i> - component.name = <i>"AuthCtrlr"</i> - variable.name = <i>"AuthorizeRemoteStart"</i> - attributeStatusInfo is absent or attributeStatusInfo.reasonCode = <i>InvalidValue</i> (not required) | |
| <p>* Step 6:</p> <p>Message: SetVariablesResponse has</p> <p>SetVariableResultType</p> <ul style="list-style-type: none"> - attributeStatus = <i>Rejected/Accepted</i> - attributeType = <i>Actual</i> - component.name = <i>"SmartChargingCtrlr"</i> - variable.name = <i>"LimitChangeSignificance"</i> - attributeStatusInfo is absent or attributeStatusInfo.reasonCode = <i>InvalidValue</i> (not required) | |
| <p>* Step 8:</p> <p>Message: SetVariablesResponse has</p> <p>SetVariableResultType</p> <ul style="list-style-type: none"> - attributeStatus = <i>Rejected/Accepted</i> - attributeType = <i>Actual</i> - component.name = <i>"ClockCtrlr"</i> - variable.name = <i>"NextTimeOffsetTransitionDateTime"</i> - attributeStatusInfo is absent or attributeStatusInfo.reasonCode = <i>InvalidValue</i> (not required) | |
| <p>Post scenario validations:</p> <p>N/A</p> | |

TC_B_39_CS: Set Variables - Read-only

| | |
|-------------------|---|
| Test case name | Set Variables - Read-only |
| Test case Id | TC_B_39_CS |
| Use case Id(s) | B05 |
| Requirement(s) | B05.FR.09 |
| System under test | Charging Station |
| Description | The CSMS can use a SetVariablesRequest to set values from device model variables at the Charging Station. |
| Purpose | To verify whether the Charging Station can handle receiving a SetVariablesRequest for a Read-only variable. |
| Prerequisite(s) | N/A |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|---|--|
| Charging Station | CSMS |
| 2. The Charging Station responds with SetVariablesResponse | 1. Test System sends SetVariablesRequest with: - variable.name = "MessageTimeout" - variable.instance = "Default" - component.name = "OCPPCommCtrlr" - attributeType is omitted |

| |
|---|
| Tool validations |
| * Step 2: Message: SetVariablesResponse - setVariableResult[0].attributeStatus = Rejected - setVariableResult[0].component.name = "OCPPCommCtrlr" - setVariableResult[0].variable.name = "MessageTimeout" - setVariableResult[0].variable.instance = "Default" |
| Post scenario validations: N/A |

TC_B_12_CS: Get Base Report - ConfigurationInventory

| | |
|-------------------|--|
| Test case name | Get Base Report - ConfigurationInventory |
| Test case Id | TC_B_12_CS |
| Use case Id(s) | B07 |
| Requirement(s) | B07.FR.01, B07.FR.03, B07.FR.04, B07.FR.07, B07.FR.10, B07.FR.12 |
| System under test | Charging Station |
| Description | CSMS requests a ConfigurationInventory base report. |
| Purpose | To test that Charging Station supports the ConfigurationInventory base report. |
| Prerequisite(s) | N/A |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. Charging Station responds with: GetBaseReportResponse | 1. Test System sends GetBaseReportRequest with: - requestId = <Generated requestId> - reportBase = <i>ConfigurationInventory</i> |
| 3. Charging Station responds with: NotifyReportRequest | 4. Test System sends NotifyReportResponse |
| Step 3 and 4 are repeated as often as needed to report all configuration variables. | |

| | |
|---|---|
| Tool validations | |
| * Step 2: Message: GetBaseReportResponse - status = <i>Accepted</i> - statusInfo is absent or statusInfo.reasonCode = <i>"NoError"</i> | |
| * Step 3: Message: NotifyReportRequest - requestId = <Generated requestId> - generatedAt = <timestamp at charging station> - seqNo = 0 - if variableCharacteristics.dataType = <i>OptionList, SequenceList</i> or <i>MemberList</i> then valuesList must be provided. | |
| while tbc = <i>true</i> | Expect NotifyReportRequest - seqNo is incremented by 1 |
| Post scenario validations: Check for all received variables: - variableCharacteristics are present - mutability = <i>ReadWrite</i> or <i>WriteOnly</i> Validate that as a minimum the required writable variables in section "Referenced Components and Variables" are reported, that are relevant to each functional block that has been implemented. | |

TC_B_13_CS: Get Base Report - FullInventory

| | |
|-------------------|---|
| Test case name | Get Base Report - FullInventory |
| Test case Id | TC_B_13_CS |
| Use case Id(s) | B07 |
| Requirement(s) | B07.FR.01, B07.FR.03, B07.FR.04, B07.FR.08, B07.FR.10, B07.FR.12 |
| System under test | Charging Station |
| Description | CSMS requests a FullInventory base report. |
| Purpose | To test that Charging Station supports the FullInventory base report. |
| Prerequisite(s) | N/A |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. Charging Station responds with: GetBaseReportResponse | 1. Test System sends GetBaseReportRequest with: - requestId = <Generated requestId> - reportBase = FullInventory |
| 3. Charging Station responds with: NotifyReportRequest | 4. Test System sends NotifyReportResponse |
| Step 3 and 4 are repeated as often as needed to report all configuration variables. | |

| | |
|---|---|
| Tool validations | |
| * Step 2: Message: GetBaseReportResponse - status = <i>Accepted</i> - statusInfo is absent or statusInfo.reasonCode = <i>"NoError"</i> | |
| * Step 3: Message: NotifyReportRequest - requestId = <i><Generated requestId></i> - generatedAt = <i><timestamp at charging station></i> - seqNo = 0 - if variableCharacteristics.dataType = <i>OptionList, SequenceList</i> or <i>MemberList</i> then valuesList must be provided. | |
| while tbc = <i>true</i> | Expect NotifyReportRequest - seqNo is incremented by 1 |
| Post scenario validations: Check for all received variables: - variableCharacteristics are present Validate that as a minimum the required variables mentioned in section "Charging Infrastructure Related" are reported as well as the required variables in section "Referenced Components and Variables", that are relevant to each functional block that has been implemented. | |

TC_B_14_CS: Get Base Report - SummaryInventory

| | |
|-------------------|---|
| Test case name | Get Base Report - SummaryInventory |
| Test case Id | TC_B_14_CS |
| Use case Id(s) | B07 |
| Requirement(s) | B07.FR.01, B07.FR.03, B07.FR.04, B07.FR.09, B07.FR.10, B07.FR.12 |
| System under test | Charging Station |
| Description | CSMS requests a SummaryInventory base report. |
| Purpose | To test that Charging Station supports the SummaryInventory base report. |
| Prerequisite(s) | Charging Station implementation supports the optional SummaryInventory report |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. Charging Station responds with: GetBaseReportResponse | 1. Test System sends GetBaseReportRequest with: - requestId = <Generated requestId> - reportBase = <i>SummaryInventory</i> |
| 3. Charging Station responds with: NotifyReportRequest | 4. Test System sends NotifyReportResponse |
| Step 3 and 4 are repeated as often as needed to report all configuration variables. | |

| | |
|---|---|
| Tool validations | |
| * Step 2: Message: GetBaseReportResponse - status = <i>Accepted</i> - statusInfo is absent or statusInfo.reasonCode = <i>"NoError"</i> | |
| * Step 3: Message: NotifyReportRequest - requestId = <Generated requestId> - generatedAt = <timestamp at charging station> - seqNo = 0 | |
| while tbc = <i>true</i> | Expect NotifyReportRequest - seqNo is incremented by 1 |
| Post scenario validations: Check for all received variables: - variableCharacteristics are present - if variableCharacteristics.dataType = <i>OptionList, SequenceList</i> or <i>MemberList</i> then valuesList must be provided. Result must be a report that lists Components/Variables relating to the Charging Station's current charging availability, and to any existing problem conditions. - For the Charging Station Component: AvailabilityState - For each EVSE Component: AvailabilityState - For each Connector Component: AvailabilityState (if known and different from EVSE). - For all Components in an abnormal State: - Problem, Tripped, Overload, Fallback variables. - Any other diagnostically relevant Variables of the Components. | |

TC_B_15_CS: Get Base Report - Not Supported base report

| | |
|-------------------|--|
| Test case name | Get Base Report - Not Supported base report |
| Test case Id | TC_B_15_CS |
| Use case Id(s) | B07 |
| Requirement(s) | B07.FR.02 |
| System under test | Charging Station |
| Description | CSMS requests a base report that is not supported. |
| Purpose | To test that Charging Station returns NotSupported when a SummaryInventory base report is requested, but Charging Station does not support it. |
| Prerequisite(s) | Charging Station implementation does not support the optional SummaryInventory report. |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State(s):

N/a

Main (Test scenario)

| Charging Station | CSMS |
|--|--|
| 2. Charging Station responds with: GetBaseReportResponse | 1. Test System sends GetBaseReportRequest with: - requestId = <Generated requestId> - reportBase = SummaryInventory |
| Note(s): - Test System waits to make sure CS does not send a NotifyReportRequest | |

Tool validations

* Step 2:

Charging Station responds with:

GetBaseReportResponse with:

- **status** = NotSupported
- **statusInfo** is absent or **statusInfo.reasonCode** = "UnsupportedParam"

Post scenario validations:

N/A

TC_B_16_CS: Get Custom Report - with component criteria

| | |
|-------------------|--|
| Test case name | Get Custom Report - with component criteria |
| Test case Id | TC_B_16_CS |
| Use case Id(s) | B08 |
| Requirement(s) | B08.FR.01, B08.FR.03, B08.FR.04, B08.FR.07, B08.FR.09, B08.FR.10, B08.FR.12, B08.FR.13, B08.FR.14, B06.FR.09 |
| System under test | Charging Station |
| Description | CSMS requests a custom report based on a set of component criteria. |
| Purpose | To test that Charging Station supports a custom report query. |
| Prerequisite(s) | Charging Station has implemented custom reporting (use case B08). |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|--|--|
| Charging Station | CSMS |
| | 1. Test System sends GetReportRequest with: - requestId = <Generated requestId> - componentCriteria = { Enabled } |
| 2. Charging Station responds: GetReportResponse | |
| 3. Charging Station sends NotifyReportRequest with: - reportData for all components that have the variable "Enabled" set to "true". | 4. Test System responds with NotifyReportResponse |
| Steps 3 and 4 may be repeated multiple times until everything has been reported. | |
| Test System sends a <i>GetVariables</i> for all variables to match results. | |
| 6. Charging Station responds with: GetVariablesResponse | 5. Test System sends GetVariablesRequest |

| Tool validations | |
|--|---|
| <p>* Step 2: +2. Message: GetReportResponse with:</p> <ul style="list-style-type: none">- status = <i>Accepted</i>- statusInfo is absent or statusInfo = <i>"NoError"</i> | |
| <p>* step 3:</p> <p>Message: NotifyReportRequest with:</p> <ul style="list-style-type: none">- requestId = <i><Generated requestId></i>- generatedAt = <i><time of generation at charging station></i>- seqNo = <i>0</i>- reportData.variableCharacteristics are present- reportData.variable.name is <i>"Enabled"</i>- reportData.variableAttribute.value = <i>"true"</i> <p><i>Note: for Enabled there will only be an Actual value.</i></p> <p><i>It does not make any sense to have a MinSet, MaxSet or Target value for them.</i></p> | |
| While tbc = <i>true</i> | <p>Message: NotifyReportRequest</p> <ul style="list-style-type: none">- seqNo is incremented by one- reportData.variableCharacteristics are present- reportData.variable.name is <i>"Enabled"</i>- reportData.variableAttribute.value = <i>"true"</i> |

| Tool validations |
|---|
| <p>* Step 6:</p> <p>For component variables where NotifyReportRequest.reportData.variableAttribute.mutability from step 3 is not <i>WriteOnly</i></p> <ul style="list-style-type: none">- attributeStatus = <i>Accepted</i>- attributeValue = <i>true</i> <p>For component variables where NotifyReportRequest.reportData.variableAttribute.mutability from step 3 is <i>WriteOnly</i></p> <ul style="list-style-type: none">- attributeStatus = <i>Rejected</i>- attributeValue = <i><omitted></i> |
| <p>Post scenario validations:</p> <p>Check that every variable, named "Enabled" that was reported in the FullInventory base report with a value of "true" is also reported by the custom report for componentCriteria = { <i>Enabled</i> }.</p> |

TC_B_17_CS: Get Custom Report - with component/variable

| | |
|-------------------|---|
| Test case name | Get Custom Report - with component/variable |
| Test case Id | TC_B_17_CS |
| Use case Id(s) | B08 |
| Requirement(s) | B08.FR.01, B08.FR.03, B08.FR.04, B08.FR.11, B08.FR.12, B08.FR.14 |
| System under test | Charging Station |
| Description | CSMS requests a custom report for AvailabilityState of EVSE #1. |
| Purpose | To test that Charging Station supports a custom report query. |
| Prerequisite(s) | Charging Station has implemented custom reporting (use case B08). |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|---|---|
| Charging Station | CSMS |
| 2. Charging Station responds: GetReportResponse | 1. Test System sends GetReportRequest with: - requestId = <Generated requestId> - componentVariable[0].component.name = "EVSE" - componentVariable[0].component.evse.id = 1 - componentVariable[0].variable.name = "AvailabilityState" |
| 3. Charging Station sends NotifyReportRequest | 4. Test System responds with NotifyReportResponse |

| |
|--|
| Tool validations |
| <p>* Step 2:</p> <p>Message: GetReportResponse with:</p> <ul style="list-style-type: none"> - status = <i>Accepted</i> - statusInfo is absent or statusInfo = "NoError" |
| <p>* step 3:</p> <p>Message: NotifyReportRequest with:</p> <ul style="list-style-type: none"> - requestId = <i>GetReportRequest requestId</i> - generatedAt = <time of generation at charging station> - seqNo = 0 - reportData.component.name = "EVSE" - reportData.component.evse.id = 1 - reportData.variable.name = "AvailabilityState" - reportData.variableCharacteristics.dataType = <i>OptionList</i> - reportData.variableCharacteristics.valuesList = "Available, Occupied, Reserved, Unavailable, Faulted" - reportData.variableAttribute.mutability = <i>ReadOnly</i> - reportData.variableAttribute.type = <i>Actual</i> <p>Note: for AvailabilityState there will only be an <i>_Actual</i> value. It does not make any sense to have a <i>MinSet</i>, <i>MaxSet</i> or <i>Target</i> value for it.</p> |
| Post scenario validations: N/A |

TC_B_18_CS: Get Custom Report - with component criteria and component/variable

| | |
|-------------------|---|
| Test case name | Get Custom Report - with component criteria and component/variable |
| Test case Id | TC_B_18_CS |
| Use case Id(s) | B08 |
| Requirement(s) | B08.FR.01, B08.FR.03, B08.FR.04, B08.FR.05, B08.FR.11, B08.FR.12, B08.FR.14, B08.FR.15 |
| System under test | Charging Station |
| Description | CSMS requests a custom report for AvailabilityState of EVSE #1 as <i>Available</i> and with <i>Problem</i> . |
| Purpose | To test that Charging Station supports a custom report query and that it takes the component criteria into account, by first request a selection that should return a value and then requesting a selection that should not return a value. |
| Prerequisite(s) | Charging Station has implemented custom reporting (use case B08). |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| Request EVSE::AvailabilityState from _Available components. Note 1: EVSE #1 must be available, because a Charging Station without EVSE is useless. Note 2: Do not confuse <i>Available</i> with <i>AvailabilityState</i> ._ | |
| 2. Charging Station responds: GetReportResponse | 1. Test System sends GetReportRequest with: - requestId = <Generated requestId> - componentCriteria = { <i>Available</i> } - componentVariable[0].component.name = "EVSE" - componentVariable[0].component.evse.id = 1 - componentVariable[0].variable.name = "AvailabilityState" |
| 3. Charging Station sends NotifyReportRequest | 4. Test System responds with NotifyReportResponse |
| Request EVSE::AvailabilityState from _Problem components. Note 1: Assuming EVSE #1 does not have <i>Problem</i> variable set._ | |
| 6. Charging Station responds: GetReportResponse | 5. Test System sends GetReportRequest with: - requestId = <Generated requestId> - componentCriteria[0] = <i>Problem</i> - componentVariable[0].component.name = "EVSE" - componentVariable[0].component.evse.id = 1 - componentVariable[0].variable.name = "AvailabilityState" |

| |
|---|
| Tool validations |
| * Step 2: Message: GetReportResponse with: - status = <i>Accepted</i> - statusInfo is absent or statusInfo = "NoError" |

| Tool validations |
|---|
| <p>* step 3:</p> <p>Message: NotifyReportRequest with:</p> <ul style="list-style-type: none">- requestId = <i>GetReportRequest requestid</i>- generatedAt = <time of generation at charging station>- seqNo = 0- reportData.component.name = "EVSE"- reportData.component.evse.id = 1- reportData.variable.name = "AvailabilityState"- reportData.variableCharacteristics.dataType = <i>OptionList</i>- reportData.variableCharacteristics.valuesList = "Available, Occupied, Reserved, Unavailable, Faulted"- reportData.variableAttribute.mutability = <i>ReadOnly</i>- reportData.variableAttribute.type = <i>Actual</i> <p>Note: for AvailabilityState there will only be an <i>_Actual</i> value. It does not make any sense to have a <i>MinSet</i>, <i>MaxSet</i> or <i>Target</i> value for it.</p> |
| <p>* Step 6:</p> <p>Message: GetReportResponse with:</p> <ul style="list-style-type: none">- status = <i>EmptyResultSet</i>- statusInfo is absent or statusInfo.reasonCode = "NotFound" |
| <p>Post scenario validations:</p> <p>N/A</p> |

TC_B_19_CS: Get Custom Report - for unknown component criteria

NOTE

Since ComponentCriterionEnum is defined as enumeration, this will most likely already be caught by the JSON parser.

| | |
|-------------------|--|
| Test case name | Get Custom Report - for unknown component criteria |
| Test case Id | TC_B_19_CS |
| Use case Id(s) | B08 |
| Requirement(s) | B08.FR.02 |
| System under test | Charging Station |
| Description | CSMS sends a GetReport with an invalid value in componentCriteria . |
| Purpose | To test that Charging Station returns a <i>NotSupported</i> return code in response to an invalid value for componentCriteria . |
| Prerequisite(s) | The Charging Station has one or more not supported componentCriteria. |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. Charging Station responds with: GetReportResponse | 1. Test System sends GetReportRequest with: - requestId = <Generated requestId> - componentCriteria = { Available, <Configured Unsupported componentCriteria> } - *componentVariable is absent |

| |
|---|
| Tool validations |
| * Step 2 Message: GetReportResponse - status = <i>NotSupported</i> - statusInfo is absent or statusInfo.reasonCode = " <i>UnsupportedParam</i> " or statusInfo.reasonCode = " <i>InvalidValue</i> " |
| Post scenario validations: N/A |

TC_B_20_CS: Reset Charging Station - Without ongoing transaction - OnIdle

| | |
|-------------------|--|
| Test case name | Reset Charging Station - Without ongoing transaction - OnIdle |
| Test case Id | TC_B_20_CS |
| Use case Id(s) | B11 |
| Requirement(s) | B11.FR.01, B11.FR.03, B11.FR.04, B01.FR.03 |
| System under test | Charging Station |
| Description | This test case covers how the CSMS can request the Charging Station to reset itself by sending a ResetRequest without any ongoing transaction. This could for example be necessary if the Charging Station is not functioning correctly. |
| Purpose | To verify if the Charging Station is able to perform the reset mechanism as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a ResetResponse | 1. The Test System sends a ResetRequest with type OnIdle |
| <u>Note(s):</u> - Charging Station reboots | |
| 3. The Charging Station sends a BootNotificationRequest | 4. The Test System responds with a BootNotificationResponse with status Accepted |
| 5. The Charging Station notifies the CSMS about the current state of all connectors. | 6. The Test System responds accordingly. |

| |
|--|
| Tool validations |
| <p>* Step 2: Message ResetResponse - status Accepted</p> <p>* Step 5: Message: StatusNotificationRequest - connectorStatus Available Message: NotifyEventRequest - eventData[0].trigger Delta - eventData[0].actualValue "Available" - eventData[0].component.name "Connector" - eventData[0].variable.name "AvailabilityState"</p> |
| <p>Post scenario validations: - A message to report the state of a connector has been received for all connectors.</p> |

TC_B_21_CS: Reset Charging Station - With Ongoing Transaction - OnIdle

| | |
|-------------------|--|
| Test case name | Reset Charging Station - With Ongoing Transaction - OnIdle |
| Test case Id | TC_B_21_CS |
| Use case Id(s) | B12 |
| Requirement(s) | B12.FR.01, B12.FR.03 |
| System under test | Charging Station |
| Description | <p>This test case covers how the CSMS can remotely request the Charging Station to reset itself by sending a ResetRequest during a transaction. When ResetRequest "OnIdle" is send the charging stations schedules a reboot after all transactions are stopped.</p> <p>This could for example be necessary if the Charging Station is not functioning correctly.</p> |
| Purpose | To verify if the Charging Station is able to perform the reset mechanism while there is an ongoing transaction as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a ResetResponse | 1. The Test System sends a ResetRequest with type OnIdle |
| 3. Execute Reusable State <i>StopAuthorized</i> | |
| 4. Execute Reusable State <i>EVConnectedPostSession</i> | |
| 5. Execute Reusable State <i>EVDisconnected</i> | |
| <u>Notes(s)</u> : Steps 4 and 5 will only be executed if TxStartPoint does not contain: EnergyTransferStarted, DataSigned, PowerPathClosed, or Authorized | |
| 6. Execute Reusable State <i>ParkingBayUnoccupied</i> | |
| <u>Notes(s)</u> : Step 6 will only be executed if TxStartPoint does not contain: EnergyTransferStarted, DataSigned, PowerPathClosed, Authorized, or EVConnected | |
| 7. The Charging Station sends a BootNotificationRequest | 8. The Test System responds with a BootNotificationResponse |
| 9. The Charging Station notifies the CSMS about the current state of all connectors. | 10. The Test System responds accordingly. |
| 11. The Charging Station sends a SecurityEventNotificationRequest | 12. The Test System responds with a SecurityEventNotificationResponse |

| Tool validations |
|---|
| <p>* Step 2: Message ResetResponse - status <i>Scheduled</i></p> <p>* Step 7: Message BootNotificationRequest - reason <i>ScheduledReset</i></p> <p>* Step 9: Message: StatusNotificationRequest - If the transaction was stopped at step 3, then — for the connector involved in the transaction: connectorStatus <i>Occupied</i> — for other connectors of this EVSE: connectorStatus <i>Available</i> or <i>Unavailable</i> Else connectorStatus <i>Available</i> Message: NotifyEventRequest - eventData[0].trigger <i>Delta</i> - eventData[0].component.name <i>"Connector"</i> - eventData[0].variable.name <i>"AvailabilityState"</i> - If the transaction was stopped at step 3, then — for the connector involved in the transaction: eventData[0].actualValue <i>Occupied</i> — for other connectors of this EVSE: eventData[0].actualValue <i>Available</i> or <i>Unavailable</i> - Else eventData[0].actualValue <i>"Available"</i></p> <p>* Step 11: Message: SecurityEventNotificationRequest - type <i>StartupOfTheDevice</i> or <i>ResetOrReboot</i></p> |
| <p>Post scenario validations: - A message to report the state of a connector has been received for all connectors.</p> |

TC_B_22_CS: Reset Charging Station - With Ongoing Transaction - Immediate

| | |
|-------------------|--|
| Test case name | Reset Charging Station - With Ongoing Transaction - Immediate |
| Test case Id | TC_B_22_CS |
| Use case Id(s) | B12 |
| Requirement(s) | B12.FR.02, B12.FR.04, E07.FR.03, B01.FR.03 |
| System under test | Charging Station |
| Description | <p>This test case covers how the CSMS can remotely request the Charging Station to reset itself by sending a ResetRequest during a transaction. When ResetRequest "Immediate" is send the charging stations will try to stop all transactions before rebooting.</p> <p>This could for example be necessary if the Charging Station is not functioning correctly.</p> |
| Purpose | To verify if the Charging Station is able to perform the reset mechanism as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a ResetResponse | 1. The Test System sends a ResetRequest with type Immediate |
| 3. The Charging Station sends a TransactionEventRequest | 4. The Test System responds with a TransactionEventResponse |
| <u>Note(s):</u> - Charging Station reboots | |
| 5. The Charging Station sends a BootNotificationRequest | 6. The Test System responds with a BootNotificationResponse with status Accepted |
| 7. The Charging Station notifies the CSMS about the current state of all connectors. | 8. The Test System responds accordingly. |

| Tool validations |
|--|
| <p>* Step 2:</p> <p>Message ResetResponse</p> <ul style="list-style-type: none"> - status <i>Accepted</i> <p>* Step 3:</p> <p>Message TransactionEventRequest</p> <ul style="list-style-type: none"> - eventType <i>Ended</i> - triggerReason <i>ResetCommand</i> - transactionInfo.chargingState <i>EVConnected</i> - transactionInfo.stoppedReason <i>ImmediateReset</i> - idToken must be omitted <p>* Step 5:</p> <p>Message BootNotificationRequest</p> <ul style="list-style-type: none"> - reason <i>RemoteReset</i> <p>* Step 7:</p> <p>For <Configured connectorId>:</p> <p>Message: StatusNotificationRequest</p> <ul style="list-style-type: none"> - for the connector involved in the transaction: connectorStatus <i>Occupied</i> - for other connectors of the same EVSE: connectorStatus <i>Available</i> or <i>Unavailable</i> <p>Message: NotifyEventRequest</p> <ul style="list-style-type: none"> - eventData[0].trigger <i>Delta</i> - for the connector involved in the transaction: eventData[0].actualValue <i>Occupied</i> - for other connectors of the same EVSE: eventData[0].actualValue <i>Available</i> or <i>Unavailable</i> - eventData[0].component.name <i>"Connector"</i> - eventData[0].variable.name <i>"AvailabilityState"</i> <p>For <Other connector(s)>:</p> <p>Message: StatusNotificationRequest</p> <ul style="list-style-type: none"> - connectorStatus <i>Available</i> <p>Message: NotifyEventRequest</p> <ul style="list-style-type: none"> - eventData[0].trigger <i>Delta</i> - eventData[0].actualValue <i>"Available"</i> - eventData[0].component.name <i>"Connector"</i> - eventData[0].variable.name <i>"AvailabilityState"</i> |
| <p>Post scenario validations:</p> <ul style="list-style-type: none"> - A message to report the state of a connector has been received for all connectors. |

TC_B_23_CS: Reset Charging Station - Unavailable persists reset

| | |
|-------------------|--|
| Test case name | Reset Charging Station - Unavailable persists reset |
| Test case Id | TC_B_23_CS |
| Use case Id(s) | B11 |
| Requirement(s) | B11.FR.01, B11.FR.02, B11.FR.03, B11.FR.04, B01.FR.03 |
| System under test | Charging Station |
| Description | This test case covers how the CSMS can request the Charging Station to reset itself by sending a ResetRequest without any ongoing transaction and with the status of Inoperative . This could for example be necessary if the Charging Station is not functioning correctly. |
| Purpose | To verify if the Charging Station is able to perform the reset mechanism as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: <i>Unavailable</i> for <Configured connectorId> |
| Reusable State(s): N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a ResetResponse | 1. The Test System sends a ResetRequest with type OnIdle |
| Note(s): - The Charging Station reboots | |
| 3. The Charging Station sends a BootNotificationRequest | 4. The Test System responds with a BootNotificationResponse with status Accepted |
| 5. The Charging Station notifies the CSMS about the current state of all connectors. | 6. The Test System responds accordingly. |

| Tool validations |
|--|
| <p>* Step 2:</p> <p>Message ResetResponse</p> <ul style="list-style-type: none">- status <i>Accepted</i> <p>* Step 3:</p> <p>Message BootNotificationRequest</p> <p>reason <i>RemoteReset</i></p> <p>* Step 5:</p> <p>For <Configured connectorId>:</p> <p>Message: StatusNotificationRequest</p> <ul style="list-style-type: none">- connectorStatus <i>Unavailable</i> <p>Message: NotifyEventRequest</p> <ul style="list-style-type: none">- eventData[0].trigger <i>Delta</i>- eventData[0].actualValue <i>"Unavailable"</i>- eventData[0].component.name <i>"Connector"</i>- eventData[0].variable.name <i>"AvailabilityState"</i> <p>For <Other connector(s)>:</p> <p>Message: StatusNotificationRequest</p> <ul style="list-style-type: none">- connectorStatus <i>Available</i> <p>Message: NotifyEventRequest</p> <ul style="list-style-type: none">- eventData[0].trigger <i>Delta</i>- eventData[0].actualValue <i>"Available"</i>- eventData[0].component.name <i>"Connector"</i>- eventData[0].variable.name <i>"AvailabilityState"</i> |
| <p>Post scenario validations:</p> <ul style="list-style-type: none">- A message to report the state of a connector has been received for all connectors. |

TC_B_24_CS: Reset Charging Station - Reserved persists reset

| | |
|-------------------|---|
| Test case name | Reset Charging Station - Reserved persists reset |
| Test case Id | TC_B_24_CS |
| Use case Id(s) | B11 |
| Requirement(s) | B11.FR.01, B11.FR.03, B11.FR.04, B11.FR.05, B01.FR.03 |
| System under test | Charging Station |
| Description | This test case covers how the CSMS can request the Charging Station to reset itself by sending a ResetRequest without any ongoing transaction and with the status of Reserved . This could for example be necessary if the Charging Station is not functioning correctly. |
| Purpose | To verify if the CSMS is able to perform the reset mechanism as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: <i>Reserved</i> for <Configured connectorId> |
| Reusable State(s): N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a ResetResponse | 1. The Test System sends a ResetRequest with type Immediate |
| <u>Note(s):</u> - The Charging Station reboots | |
| 3. The Charging Station sends a BootNotificationRequest | 4. The Test System responds with a BootNotificationResponse with status Accepted |
| 5. The Charging Station notifies the CSMS about the current state of all connectors. | 6. The Test System responds accordingly. |

| Tool validations |
|--|
| <p>* Step 2:</p> <p>Message ResetResponse</p> <ul style="list-style-type: none">- status <i>Accepted</i> <p>* Step 3:</p> <p>Message BootNotificationRequest</p> <p>reason <i>RemoteReset</i></p> <p>* Step 5:</p> <p>For <Configured connectorId>:</p> <p>Message: StatusNotificationRequest</p> <ul style="list-style-type: none">- connectorStatus <i>Reserved</i> <p>Message: NotifyEventRequest</p> <ul style="list-style-type: none">- eventData[0].trigger <i>Delta</i>- eventData[0].actualValue <i>"Reserved"</i>- eventData[0].component.name <i>"Connector"</i>- eventData[0].variable.name <i>"AvailabilityState"</i> <p>For <Other connector(s)>:</p> <p>Message: StatusNotificationRequest</p> <ul style="list-style-type: none">- connectorStatus <i>Available</i> <p>Message: NotifyEventRequest</p> <ul style="list-style-type: none">- eventData[0].trigger <i>Delta</i>- eventData[0].actualValue <i>"Available"</i>- eventData[0].component.name <i>"Connector"</i>- eventData[0].variable.name <i>"AvailabilityState"</i> |
| <p>Post scenario validations:</p> <ul style="list-style-type: none">- A message to report the state of a connector has been received for all connectors. |

TC_B_41_CS: Reset Charging Station - With multiple ongoing transactions - OnIdle

| | |
|-------------------|--|
| Test case name | Reset Charging Station - With multiple ongoing transactions - OnIdle |
| Test case Id | TC_B_41_CS |
| Use case Id(s) | B12 |
| Requirement(s) | B12.FR.01, B12.FR.03, E07.FR.03 |
| System under test | Charging Station |
| Description | <p>This test case covers how the CSMS can remotely request the Charging Station to reset itself by sending a ResetRequest during a transaction. When ResetRequest "OnIdle" is send the charging stations schedules a reboot after all transactions are stopped.</p> <p>This could for example be necessary if the Charging Station is not functioning correctly.</p> |
| Purpose | To verify if the Charging Station is able to perform the reset mechanism while there are multiple ongoing transactions as described at the OCPP specification. |
| Prerequisite(s) | The Charging Station has more than one EVSE. |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> for EVSE.id = 1 State is <i>EnergyTransferStarted</i> for EVSE.id = 2 |

| Main (Test scenario) | |
|--|--|
| Charging Station | CSMS |
| | 1. The Test System sends a ResetRequest with type OnIdle |
| 2. The Charging Station responds with a ResetResponse | |
| 3. Execute Reusable State <i>StopAuthorized</i> for EVSE.id = 1 | |
| 4. Execute Reusable State <i>EVConnectedPostSession</i> for EVSE.id = 1 | |
| 5. Execute Reusable State <i>EVDisconnected</i> for EVSE.id = 1 | |
| 6. Execute Reusable State <i>ParkingBayUnoccupied</i> for EVSE.id = 1 | |
| 7. Execute Reusable State <i>StopAuthorized</i> for EVSE.id = 2 | |
| 8. Execute Reusable State <i>EVConnectedPostSession</i> for EVSE.id = 2 | |
| <u>Note(s):</u> If TxStopPoint contains one of the following values; Authorized, EnergyTransfer, PowerPathClosed, DataSigned. Then the transaction will have ended at the <i>EVConnectedPostSession</i> state AND the Charging Station will proceed with resetting itself. Proceed to step 11 Else proceed with step 9. | |
| 9. Execute Reusable State <i>EVDisconnected</i> for EVSE.id = 2 | |
| <u>Note(s):</u> If TxStopPoint contains the value EVConnected. Then the transaction will have ended at the <i>EVDisconnected</i> state AND the Charging Station will proceed with resetting itself. Proceed to step 11 Else proceed with step 10 | |
| 10. Execute Reusable State <i>ParkingBayUnoccupied</i> for EVSE.id = 2 | |
| <u>Note(s):</u> The transaction will end at this state, if it was not ended at an earlier state. Proceed to step 11. | |
| 11. The Charging Station sends a BootNotificationRequest | 12. The Test System responds with a BootNotificationResponse |

| Main (Test scenario) | |
|--|---|
| 13. The Charging Station notifies the CSMS about the current state of all connectors. | 14. The Test System responds accordingly. |
| Tool validations | |
| <p>* Step 2: Message ResetResponse - status <i>Scheduled</i></p> <p>* Step 11: Message BootNotificationRequest - reason <i>ScheduledReset</i></p> <p>* Step 13: Message: StatusNotificationRequest - If the transaction was stopped at step 3, then – for the connector involved in the transaction: connectorStatus <i>Occupied</i> – for other connectors of this EVSE: connectorStatus <i>Available</i> or <i>Unavailable</i> - Else connectorStatus <i>Available</i></p> <p>Message: NotifyEventRequest - eventData[0].trigger <i>Delta</i> - eventData[0].component.name <i>"Connector"</i> - eventData[0].variable.name <i>"AvailabilityState"</i> - If the transaction was stopped at step 3, then – for the connector involved in the transaction: eventData[0].actualValue <i>Occupied</i> – for other connectors of this EVSE: eventData[0].actualValue <i>Available</i> or <i>Unavailable</i> - Else eventData[0].actualValue <i>"Available"</i></p> | |
| <p>Post scenario validations: - A message to report the state of a connector has been received for all connectors.</p> | |

TC_B_25_CS: Reset EVSE - Without ongoing transaction

| | |
|-------------------|---|
| Test case name | Reset EVSE - Without ongoing transaction |
| Test case Id | TC_B_25_CS |
| Use case Id(s) | B11 |
| Requirement(s) | B11.FR.01, B11.FR.08, B11.FR.10 |
| System under test | Charging Station |
| Description | This test case covers how the CSMS can request the Charging Station to reset an EVSE by sending a ResetRequest without any ongoing transaction. This could for example be necessary if the Charging Station is not functioning correctly. |
| Purpose | To verify if the Charging Station is able to perform the reset mechanism as described at the OCPP specification. |
| Prerequisite(s) | Individual resetting EVSE supported |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a ResetResponse | 1. The Test System sends a ResetRequest with type OnIdle and <i>*evseld*<Configured evseld></i> |
| <u>Note(s)</u> : - <i><Configured evseld> reboots</i> | |

| |
|--|
| Tool validations |
| * Step 2: Message ResetResponse - status <i>Accepted</i> |
| Post scenario validations: - N/a |

TC_B_26_CS: Reset EVSE - With Ongoing Transaction - OnIdle

| | |
|-------------------|---|
| Test case name | Reset EVSE - With Ongoing Transaction - OnIdle |
| Test case Id | TC_B_26_CS |
| Use case Id(s) | B12 |
| Requirement(s) | B12.FR.01, B12.FR.07 |
| System under test | Charging Station |
| Description | <p>This test case covers how the CSMS can remotely request the Charging Station to reset an EVSE by sending a ResetRequest during a transaction. When ResetRequest "OnIdle" is send the charging stations schedules a reboot after all transactions are stopped.</p> <p>This could for example be necessary if the Charging Station is not functioning correctly.</p> |
| Purpose | To verify if the Charging Station is able to perform the reset mechanism as described at the OCPP specification. |
| Prerequisite(s) | Individual resetting EVSE supported |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a ResetResponse | 1. The Test System sends a ResetRequest with type OnIdle and evseld <Configured evseld> |
| 3. Execute Reusable State <i>StopAuthorized</i> | |
| 4. Execute Reusable State <i>EVConnectedPostSession</i> | |
| 5. Execute Reusable State <i>EVDDisconnected</i> | |
| 6. Execute Reusable State <i>ParkingBayUnoccupied</i> | |

| |
|---|
| Tool validations |
| * Step 2: Message ResetResponse - status <i>Scheduled</i> |
| Post scenario validations: N/a |

TC_B_27_CS: Reset EVSE - With Ongoing Transaction - Immediate

| | |
|-------------------|---|
| Test case name | Reset EVSE - With Ongoing Transaction - Immediate |
| Test case Id | TC_B_27_CS |
| Use case Id(s) | B12 |
| Requirement(s) | B12.FR.02, B12.FR.08, E07.FR.03 |
| System under test | Charging Station |
| Description | <p>This test case covers how the CSMS can remotely request the Charging Station to reset an EVSE by sending a ResetRequest during a transaction. When ResetRequest "Immediate" is send the charging stations will try to stop all transactions before rebooting.</p> <p>This could for example be necessary if the Charging Station is not functioning correctly.</p> |
| Purpose | To verify if the Charging Station is able to perform the reset mechanism as described at the OCPP specification. |
| Prerequisite(s) | Individual resetting EVSE supported |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> |

| Main (Test scenario) | |
|--|--|
| Charging Station | CSMS |
| 2. The Charging Station responds with a ResetResponse | 1. The Test System sends a ResetRequest with type Immediate and <i>*evseld*<Configured evseld></i> |
| 3. The Charging Station sends a TransactionEventRequest | 4. The Test System responds with a TransactionEventResponse |
| <u>Note(s):</u> - The EVSE reboots | |

| |
|---|
| Tool validations |
| * Step 2: Message ResetResponse - status <i>Accepted</i> * Step 3: Message TransactionEventRequest - eventType <i>Ended</i> - triggerReason <i>ResetCommand</i> - transactionInfo.chargingState <i>EVConnected</i> - transactionInfo.stoppedReason <i>ImmediateReset</i> |
| Post scenario validations: - N/a |

TC_B_28_CS: Reset EVSE - Not Supported

| | |
|--------------------------|--|
| Test case name | Reset EVSE - Not Supported |
| Test case Id | TC_B_28_CS |
| Use case Id(s) | B11, B12 |
| Requirement(s) | B11.FR.01, B11.FR.09, B12.FR.01, B12.FR.09 |
| System under test | Charging Station |
| Description | This test case covers how the CSMS can request the Charging Station to reset an EVSE by sending a ResetRequest while it is not supported by the Charging Station. |
| Purpose | To verify if the Charging Station is able to perform the reset mechanism as described at the OCPP specification. |
| Prerequisite(s) | Charging Station does not support resetting individual EVSE |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a ResetResponse | 1. The Test System sends a ResetRequest with type OnIdle and <i>*evseld*<Configured evseld></i> |

| |
|--|
| Tool validations |
| * Step 2: Message ResetResponse - status <i>Rejected</i> |
| Post scenario validations: - N/a |

TC_B_29_CS: Reset EVSE - With ongoing transaction - Not Supported

| | |
|-------------------|--|
| Test case name | Reset EVSE - With ongoing transaction - Not Supported |
| Test case Id | TC_B_29_CS |
| Use case Id(s) | B11 |
| Requirement(s) | B12.FR.01, B12.FR.09 |
| System under test | Charging Station |
| Description | This test case covers how the CSMS can request the Charging Station to reset an EVSE by sending a ResetRequest with ongoing transaction while it is not supported by the Charging Station. |
| Purpose | To verify if the Charging Station is able to perform the reset mechanism as described at the OCPP specification. |
| Prerequisite(s) | Charging Station does not support resetting individual EVSE |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a ResetResponse | 1. The Test System sends a ResetRequest with type OnIdle and evseld <Configured evseld> |

| |
|--|
| Tool validations |
| * Step 2: Message ResetResponse - status <i>Rejected</i> |
| Post scenario validations: - N/a |

TC_B_43_CS: Set new NetworkConnectionProfile - Rejected

| | |
|-------------------|--|
| Test case name | Set new NetworkConnectionProfile - Rejected |
| Test case Id | TC_B_43_CS |
| Use case Id(s) | B09 |
| Requirement(s) | B09.FR.02 |
| System under test | Charging Station |
| Description | The CSMS updates the connection details on the Charging Station. For instance in preparation of a migration to a new CSMS. |
| Purpose | To verify if the Charging Station is able to reject when the CSMS tries to set a network connection profile containing invalid data. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|--|--|
| Charging Station | CSMS |
| 2. The Charging Station responds with a SetNetworkProfileResponse | 1. The Test System sends a SetNetworkProfileRequest with: - configurationSlot is 999 - connectionData.messageTimeout <Configured messageTimeout> - connectionData.ocppCsmsUrl <Configured ocppCsmsUrl> - connectionData.ocppInterface <Configured ocppInterface> - connectionData.ocppVersion OCPP20 - connectionData.securityProfile <Configured securityProfile> |

| |
|--|
| Tool validations |
| <p>* Step 2:</p> <p>Message SetNetworkProfileResponse</p> <ul style="list-style-type: none"> - status <i>Rejected</i> |
| <p>Post scenario validations:</p> <ul style="list-style-type: none"> - N/a |

TC_B_45_CS: Migrate to new ConnectionProfile - Success - Same CSMS Root

| | |
|-------------------|--|
| Test case name | Migrate to new ConnectionProfile - Success - Same CSMS Root |
| Test case Id | TC_B_45_CS |
| Use case Id(s) | B09, B10 |
| Requirement(s) | B09.FR.01,B10.FR.01,B10.FR.04,B10.FR.06 |
| System under test | Charging Station |
| Description | The CSMS updates the connection details on the Charging Station. For instance in preparation of a migration to a new CSMS. |
| Purpose | To verify if the Charging Station is able to migrate to another network connection profile slot. |
| Prerequisite(s) | At least two configuration slots for networkConnectionProfiles must be supported |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a SetNetworkProfileResponse | <p>1. The Test System sends a SetNetworkProfileRequest with configurationSlot is <Configured configurationSlot> or <Configured configurationSlot2> depending on which one is already in use</p> <ul style="list-style-type: none"> - connectionData.messageTimeout <Configured messageTimeout> or <Configured messageTimeout2> - connectionData.ocppCsmsUrl <ocppCsmsUrl that is not currently active> - connectionData.ocppInterface <Configured ocppInterface> or <Configured ocppInterface2> - connectionData.ocppVersion OCPP20 - connectionData.securityProfile <Configured securityProfile> or <Configured securityProfile2> <p><u>Note(s):</u></p> <ul style="list-style-type: none"> - The Test System checks the ActiveNetworkProfile variable to see which slot is currently active. - The Test System prevents overwriting the NetworkProfile at the active slot, as this is not recommended. |
| 4. The Charging Station responds with a SetVariablesResponse | <p>3. The Test System sends a SetVariablesRequest with variable.name is "NetworkConfigurationPriority"</p> <p>component.name is "OCPPCommCtrlr"</p> <p>attributeValue is <configurationSlot set at Step 1, the other configured configurationSlot></p> |
| 6. The Charging Station responds with a ResetResponse | <p>5. The Test System sends a ResetRequest with type OnIdle</p> |
| <p>7. Execute Reusable State <i>Booted</i></p> <p><u>Note(s):</u></p> <ul style="list-style-type: none"> - The Charging Station connects to the slot configured at step 1. | |

| Tool validations |
|--|
| <div>* Step 2: Message SetNetworkProfileResponse - status <i>Accepted</i></div> <div>* Step 4: Message SetVariablesResponse - setVariableResult[0].attributeStatus <i>Accepted OR RebootRequired</i></div> <div>* Step 6: Message ResetResponse - status <i>Accepted</i></div> |
| <div>Post scenario validations: - N/a</div> |

TC_B_46_CS: Migrate to new ConnectionProfile - Fallback mechanism - Same CSMS Root

| | |
|-------------------|--|
| Test case name | Migrate to new ConnectionProfile - Fallback mechanism - Same CSMS Root |
| Test case Id | TC_B_46_CS |
| Use case Id(s) | B10 |
| Requirement(s) | B10.FR.03,B10.FR.04 |
| System under test | Charging Station |
| Description | The CSMS updates the connection details on the Charging Station. For instance in preparation of a migration to a new CSMS. |
| Purpose | To verify if the Charging Station is able to use the fallback mechanism when it is unable to connect with the first network connection profile slot. |
| Prerequisite(s) | At least two configuration slots for networkConnectionProfiles must be supported |

| |
|---|
| Before (Preparations) |
| Configuration State: OCPPCommCtrlr.NetworkProfileConnectionAttempts is 2 |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a SetNetworkProfileResponse | <p>1. The Test System sends a SetNetworkProfileRequest with configurationSlot is <i><Configured configurationSlot></i> or <i><Configured configurationSlot2></i> depending on which one is already in use</p> <ul style="list-style-type: none"> - connectionData.messageTimeout <i><Configured messageTimeout></i> or <i><Configured messageTimeout2></i> - connectionData.ocppCsmsUrl <i><ocppCsmsUrl that is not currently active></i> - connectionData.ocppInterface <i><Configured ocppInterface></i> or <i><Configured ocppInterface2></i> - connectionData.ocppVersion <i>OCPP20</i> - connectionData.securityProfile <i><Configured securityProfile></i> or <i><Configured securityProfile2></i> <p><u>Note(s):</u></p> <ul style="list-style-type: none"> - The Test System checks the <i>ActiveNetworkProfile</i> variable to see which slot is currently active. - The Test System prevents overwriting the <i>NetworkProfile</i> at the active slot, as this is not recommended. |
| 4. The Charging Station responds with a SetVariablesResponse | <p>3. The Test System sends a SetVariablesRequest with variable.name is "NetworkConfigurationPriority"</p> <p>component.name is "OCPPCommCtrlr"</p> <p>attributeValue is <i><configurationSlot set at Step 1, the other configured configurationSlot></i></p> |
| 6. The Charging Station responds with a ResetResponse | <p>5. The Test System sends a ResetRequest with type <i>OnIdle</i></p> |
| | <p>7. The Test System will NOT respond to the two connection request from the Charging Station from the first connectionSlot.</p> |
| | <p>8. The Test System will accept the connection request from the Charging Station from the second connectionSlot.</p> |

| |
|--|
| Main (Test scenario) |
| <u>Note(s)</u> : Set the <Configured Long Operation Time Out> so that Steps 7 and 8 can be completed in this time period. |
| 9. Execute Reusable State <i>Booted</i> |
| <u>Note(s)</u> : - The Charging Station connects to the second slot configured at the NetworkConfigurationPriority at step 3. |

| |
|---|
| Tool validations |
| * Step 2: Message SetNetworkProfileResponse - status <i>Accepted</i> |
| * Step 4: Message SetVariablesResponse - setVariableResult[0].attributeStatus <i>Accepted OR RebootRequired</i> |
| * Step 6: Message ResetResponse - status <i>Accepted</i> |
| Post scenario validations: - N/a |

TC_B_47_CS: Migrate to new ConnectionProfile - Fallback after NetworkProfileConnectionAttempts per NetworkConfigurationPriority failed - New CSMS Root - New CSMS

| | |
|-------------------|---|
| Test case name | Migrate to new ConnectionProfile - Fallback after NetworkProfileConnectionAttempts per NetworkConfigurationPriority failed - New CSMS Root - New CSMS |
| Test case Id | TC_B_47_CS |
| Use case Id(s) | B09,B10,M05 |
| Requirement(s) | B10.FR.07,M05.FR.15,M05.FR.16 |
| System under test | Charging Station |
| Description | The CSMS updates the connection details on the Charging Station. For instance in preparation of a migration to a new CSMS. |
| Purpose | To verify if the Charging Station is able to correctly handle the fallback mechanism in the case it fails to connect to the other CSMS. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The Charging Station supports AS-2: AdditionalRootCertificateCheck. - Configured (new) CSMS Root certificate 2 must be signed by the configured (old) CSMS Root certificate 2. - At least two configuration slots for networkConnectionProfiles must be supported |

| |
|--|
| Before (Preparations) |
| Configuration State: OCPPCommCtrlr.NetworkProfileConnectionAttempts is 1 |
| Memory State: <i>CertificateInstalled</i> for certificateType CSMSRootCertificate and certificate <Configured (new) CSMS Root certificate 2> |
| Reusable State(s): N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a SetNetworkProfileResponse | <p>1. The Test System sends a SetNetworkProfileRequest with configurationSlot is <Configured configurationSlot> or <Configured configurationSlot2> depending on which one is already in use</p> <ul style="list-style-type: none"> - connectionData.messageTimeout <Configured messageTimeout> or <Configured messageTimeout2> - connectionData.ocppCsmsUrl <ocppCsmsUrl that is not currently active> - connectionData.ocppInterface <Configured ocppInterface> or <Configured ocppInterface2> - connectionData.ocppVersion OCPP20 - connectionData.securityProfile <Configured securityProfile> or <Configured securityProfile2> <p><u>Note(s):</u></p> <ul style="list-style-type: none"> - The Test System checks the ActiveNetworkProfile variable to see which slot is currently active. - The Test System prevents overwriting the NetworkProfile at the active slot, as this is not recommended. |
| 4. The Charging Station responds with a SetVariablesResponse | <p>3. The Test System sends a SetVariablesRequest with variable.name is "NetworkConfigurationPriority" component.name is "OCPPCommCtrlr" attributeValue is <configurationSlot set at Step 1, the other configured configurationSlot></p> |
| 6. The Charging Station responds with a ResetResponse | <p>5. The Test System sends a ResetRequest with type OnIdle</p> |

| Main (Test scenario) | |
|--|---|
| <p>8. During the TLS handshake the Charging Station validates the CSMS certificate.</p> <p><u>Note(s):</u> - This connection attempt fails, because the Charging Station will use the new CSMS Root certificate to validate the CSMS certificate.</p> | <p>7. During the TLS handshake the Test System provides a CSMS certificate which is signed by the <Configured old CSMS Root certificate></p> |
| <p>9. The Charging Station switches back to the previous networkprofile configuration and validates the CSMS certificate, using the (fallback) CSMS Root certificate.</p> <p><u>Note(s):</u> - This connection attempt succeeds, because the Charging Station will now use the (old) CSMS Root certificate to validate the CSMS certificate.</p> | |
| 10. Execute Reusable State <i>Booted</i> | |
| <p>12. The Charging Station responds with a GetInstalledCertificateIdsResponse</p> | <p>11. The Test System sends a GetInstalledCertificateIdsRequest with certificateType is <i>CSMSRootCertificate</i></p> |
| Tool validations | |
| <p>* Step 6: Message ResetResponse - status <i>Accepted</i></p> <p>* Step 12: Message: GetInstalledCertificateIdsResponse - status must be <i>Accepted</i> - certificateHashDataChain must contain an entry with following values: - certificateType is <i>CSMSRootCertificate</i> - certificateHashData contains <HashData from configured old CSMS Root certificate></p> | |
| <p>Post scenario validations: - N/a</p> | |

TC_B_49_CS: Migrate to new ConnectionProfile - Fallback after NetworkProfileConnectionAttempts per NetworkConfigurationPriority failed - Same CSMS Root

| | |
|-------------------|--|
| Test case name | Migrate to new ConnectionProfile - Fallback after NetworkProfileConnectionAttempts per NetworkConfigurationPriority failed - Same CSMS Root |
| Test case Id | TC_B_49_CS |
| Use case Id(s) | B10 |
| Requirement(s) | B10.FR.03, B10.FR.07 |
| System under test | Charging Station |
| Description | The CSMS updates the connection details on the Charging Station. For instance in preparation of a migration to a new CSMS. |
| Purpose | To verify if the Charging Station is able to correctly handle the fallback mechanism in the case it fails to connect to the other CSMS. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The Charging Station supports C-47: mechanism implemented & Reconnect after NetworkProfileConnectionAttempts - At least two configuration slots for networkConnectionProfiles must be supported |

| |
|---|
| Before (Preparations) |
| Configuration State: OCPPCommCtrlr.NetworkProfileConnectionAttempts is 1 OCPPCommCtrlr.RetryBackOffRepeatTimes is 0 OCPPCommCtrlr.RetryBackOffRandomRange is 0 OCPPCommCtrlr.RetryBackOffWaitMinimum is <Configured RetryBackOffWaitMinimum> |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a SetNetworkProfileResponse | <p>1. The Test System sends a SetNetworkProfileRequest with configurationSlot is <Configured configurationSlot> or <Configured configurationSlot2> depending on which one is already in use</p> <ul style="list-style-type: none"> - connectionData.messageTimeout <Configured messageTimeout> or <Configured messageTimeout2> - connectionData.ocppCsmsUrl <ocppCsmsUrl that is not currently active> - connectionData.ocppInterface <Configured ocppInterface> or <Configured ocppInterface2> - connectionData.ocppVersion OCPP20 - connectionData.securityProfile <Configured securityProfile> or <Configured securityProfile2> <p><u>Note(s):</u></p> <ul style="list-style-type: none"> - The Test System checks the ActiveNetworkProfile variable to see which slot is currently active. - The Test System prevents overwriting the NetworkProfile at the active slot, as this is not recommended. |
| 4. The Charging Station responds with a SetVariablesResponse | <p>3. The Test System sends a SetVariablesRequest with variable.name is "NetworkConfigurationPriority"</p> <p>component.name is "OCPPCommCtrlr"</p> <p>attributeValue is <configurationSlot set at Step 1, the other configured configurationSlot></p> |

| Main (Test scenario) | |
|--|--|
| 6. The Charging Station responds with a ResetResponse | 5. The Test System sends a ResetRequest with type OnIdle |
| 7. The Charging Station tries to connect to the alternative internal Test System endpoint. <u>Note(s):</u> - Make sure to set the <Configured Long Operation Time Out> to be the time required for the CS to revert to the previous network profile configuration. | 8. The connection attempt is not accepted by the Test System. |
| 9. The Charging Station switches back to the previous networkprofile configuration and reconnects to the Test System. | 10. The connection attempt is not accepted by the Test System. |
| 11. The Charging Station waits for at least the duration of the configured RetryBackOffWaitMinimum and reconnects to the Test System. | 12. The connection attempt is accepted by the Test System. |
| 13. Execute Reusable State <i>Booted</i> | |

| Tool validations |
|--|
| * Step 6: Message ResetResponse - status <i>Accepted</i> |
| Post scenario validations: - N/a |

TC_B_50_CS: Migrate to new ConnectionProfile - Success - New CSMS Root - New CSMS

| | |
|-------------------|--|
| Test case name | Migrate to new ConnectionProfile - Success - New CSMS Root - New CSMS |
| Test case Id | TC_B_50_CS |
| Use case Id(s) | B10,M05 |
| Requirement(s) | M05.FR.13 |
| System under test | Charging Station |
| Description | The CSMS updates the connection details on the Charging Station. For instance in preparation of a migration to a new CSMS. |
| Purpose | To verify if the Charging Station is able to correctly handle migrating to the new CSMS using a new CSMS Root certificate to validate the server certificate. |
| Prerequisite(s) | - At least two configuration slots for networkConnectionProfiles must be supported AND - The Charging Station must be connected using either security profile 2 or 3. |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: <i>CertificateInstalled</i> for certificateType CSMSRootCertificate and certificate <Configured (new) CSMS Root certificate 2> |
| Reusable State(s): N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a SetNetworkProfileResponse | <p>1. The Test System sends a SetNetworkProfileRequest with configurationSlot is <Configured configurationSlot> or <Configured configurationSlot2> depending on which one is already in use</p> <ul style="list-style-type: none"> - connectionData.messageTimeout <Configured messageTimeout> or <Configured messageTimeout2> - connectionData.ocppCsmsUrl <ocppCsmsUrl that is not currently active> - connectionData.ocppInterface <Configured ocppInterface> or <Configured ocppInterface2> - connectionData.ocppVersion OCPP20 - connectionData.securityProfile <Configured securityProfile> or <Configured securityProfile2> <p><u>Note(s):</u></p> <ul style="list-style-type: none"> - The Test System checks the ActiveNetworkProfile variable to see which slot is currently active. - The Test System prevents overwriting the NetworkProfile at the active slot, as this is not recommended. |
| 4. The Charging Station responds with a SetVariablesResponse | <p>3. The Test System sends a SetVariablesRequest with variable.name is "NetworkConfigurationPriority"</p> <p>component.name is "OCPPCommCtrlr"</p> <p>attributeValue is <configurationSlot set at Step 1, the other configured configurationSlot></p> |
| 6. The Charging Station responds with a ResetResponse | <p>5. The Test System sends a ResetRequest with type OnIdle</p> |

| Main (Test scenario) | |
|---|--|
| 7. The Charging Station connects to the configured alternative internal Test System endpoint. <u>Note(s):</u> - <i>During the TLS handshake the Charging Station validates and accepts the CSMS certificate, signed by the <Configured (new) CSMS Root certificate 2>.</i> | 8. The connection attempt is accepted by the Test System. |
| 9. Execute Reusable State <i>Booted</i> | |

| Tool validations |
|--|
| * Step 6: Message ResetResponse - status <i>Accepted</i> |
| Post scenario validations: - N/a |

TC_B_51_CS: Status change during offline period - > Offline Threshold

| | |
|-------------------|---|
| Test case name | Status change during offline period - > Offline Threshold |
| Test case Id | TC_B_51_CS |
| Use case Id(s) | B04 |
| Requirement(s) | B04.FR.01 |
| System under test | Charging Station |
| Description | A Charging Station sends a notification to the CSMS to inform the CSMS about a Connector status change. This can be done in two ways. Via a StatusNotificationRequest or a NotifyEventRequest from the device model. When the Charging Station is connected again to the CSMS after having been offline, and the Charging Station was longer offline than the configured threshold, it will report the status of every connector. |
| Purpose | To verify whether the Charging Station reports the status of all connectors after having been offline for longer than the configured threshold with the configuration variable OfflineThreshold . |
| Prerequisite(s) | If the Charging Station does not have more than one EVSE, this testcase will be equal to TC_B_52_CS. |

Before (Preparations)

Configuration State:

OCPPCommCtrlr.OfflineThreshold is <Configured offlineThreshold>

OCPPCommCtrlr.RetryBackOffWaitMinimum is <Configured offlineThreshold> + 2 seconds

OCPPCommCtrlr.RetryBackOffRandomRange is 0

Memory State:

N/a

Reusable State(s):

N/a

Main (Test scenario)

| Charging Station | CSMS |
|--|--|
| | 1. The Test System closes the WebSocket connection AND does not accept a reconnect. |
| 2. <u>Manual Action</u> : Connect the EV and EVSE. | |
| | 3. The Test System accepts reconnection attempt from the Charging Station, after the configured threshold has been exceeded. |
| 4. The Charging Station notifies the CSMS about the current state of all connectors. | 5. The Test System responds accordingly. |

| Tool validations |
|--|
| <p>* Step 4:</p> <p><i>Configured EVSE/Connector:</i></p> <p>Message: StatusNotificationRequest</p> <ul style="list-style-type: none">- evseld <Configured evseld>- connectorId <Configured connectorId>- for the connector involved in the transaction: connectorStatus <i>Occupied</i>- for other connectors of the same EVSE: connectorStatus <i>Available</i> or <i>Unavailable</i> <p>Message: NotifyEventRequest</p> <ul style="list-style-type: none">- eventData[0].trigger <i>Delta</i>- eventData[0].component.name <i>"Connector"</i>- eventData[0].variable.name <i>"AvailabilityState"</i>- eventData[0].evse.id <Configured evseld>- eventData[0].connectorId <Configured connectorId>- for the connector involved in the transaction: eventData[0].actualValue <i>Occupied</i>- for other connectors of the same EVSE: eventData[0].actualValue <i>Available</i> or <i>Unavailable</i> <p><i>All other EVSE/Connector(s):</i></p> <p>Message: StatusNotificationRequest</p> <ul style="list-style-type: none">- connectorStatus <i>Available</i> <p>Message: NotifyEventRequest</p> <ul style="list-style-type: none">- eventData[0].trigger <i>Delta</i>- eventData[0].actualValue <i>"Available"</i>- eventData[0].component.name <i>"Connector"</i>- eventData[0].variable.name <i>"AvailabilityState"</i> |
| <p>Post scenario validations:</p> <p>N/a</p> |

TC_B_52_CS: Status change during offline period - < Offline Threshold

| | |
|-------------------|--|
| Test case name | Status change during offline period - < Offline Threshold |
| Test case Id | TC_B_52_CS |
| Use case Id(s) | B04 |
| Requirement(s) | B04.FR.02 |
| System under test | Charging Station |
| Description | A Charging Station sends a notification to the CSMS to inform the CSMS about a Connector status change. This can be done in two ways. Via a StatusNotificationRequest or a NotifyEventRequest from the device model. When the Charging Station is connected again to the CSMS after having been offline, and the Charging Station was shorter offline than the configured threshold, it will report the status of all connector that received a status change. |
| Purpose | To verify whether the Charging Station reports the status of connectors that received a status change after having been offline for shorter than the configured threshold with the configuration variable OfflineThreshold . |
| Prerequisite(s) | N/a |

Before (Preparations)

Configuration State:

OCPPEndpoint.OfflineThreshold is <Configured offlineThreshold>

OCPPEndpoint.RetryBackOffWaitMinimum is <Configured offlineThreshold> / 2

OCPPEndpoint.RetryBackOffRandomRange is 0

Memory State:

N/a

Reusable State(s):

N/a

Main (Test scenario)

| Charging Station | CSMS |
|--|--|
| 1. The Test System closes the WebSocket connection AND does not accept a reconnect. | |
| 2. <u>Manual Action</u> : Connect the EV and EVSE. | |
| 3. The Test System accepts reconnection attempt from the Charging Station. | |
| 4. The Charging Station notifies the CSMS about the current state of the configured connector. | 5. The Test System responds accordingly. |

Tool validations

* Step 3:

Message: **StatusNotificationRequest**

- **evseld** <Configured evseld>

- **connectorId** <Configured connectorId>

- for the connector involved in the transaction: **connectorStatus** *Occupied*

- optionally for other connectors of the same EVSE: **connectorStatus** *Unavailable*

Message: **NotifyEventRequest**

- **eventData[0].trigger** *Delta*

- **eventData[0].component.name** *"Connector"*

- **eventData[0].variable.name** *"AvailabilityState"*

- **eventData[0].evse.id** <Configured evseld>

- **eventData[0].connectorId** <Configured connectorId>

- for the connector involved in the transaction: **eventData[0].actualValue** *Occupied*

- optionally for other connectors of the same EVSE: **eventData[0].actualValue** *Unavailable*

Post scenario validations:

N/a

TC_B_53_CS: Get Base Report - Test mandatory DM variables via FullInventory

| | |
|-------------------|---|
| Test case name | Get Base Report - Test mandatory DM variables via FullInventory |
| Test case Id | TC_B_53_CS |
| Use case Id(s) | B07 |
| Requirement(s) | Chapter Referenced Components and Variables |
| System under test | Charging Station |
| Description | CSMS requests a FullInventory base report. |
| Purpose | To test that Charging Station supports all required DM variables. |
| Prerequisite(s) | N/a |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State(s):

N/a

Main (Test scenario)

| Charging Station | CSMS |
|--|--|
| <p>2. CS responds with:</p> <p>GetBaseReportResponse with status = <i>Accepted</i></p> <p>3. CS sends one or more NotifyReportRequest messages to report all its component/variables.</p> | <p>1. Test System requests a GetBaseReportRequest with:</p> <p>reportBase = <i>FullInventory</i> and</p> <p>requestId = <Generated requestId></p> <p>4. Test System responds with a NotifyReportResponse for each NotifyReportRequest</p> |

Tool validations

* Step 2:

Message: **GetBaseReportResponse** with:

- **status** = *Accepted*
- **statusInfo** is absent or **statusInfo** = *"NoError"*

* step 3:

Message: **NotifyReportRequest** with:

- **requestId** = <Generated requestId>
- **generatedAt** = <time of generation at charging station>
- **seqNo** = 0

While **tbc** = *true*Message: **NotifyReportRequest**

- **seqNo** is incremented by one

Post scenario validations:

The Test System checks that all implemented standardized components / variables are implemented correctly according to the OCPP specification:

- The components / variables that are required according to the OCPP specification are implemented.
- For each component/variable, where **variableCharacteristics.dataType** is set to **OptionList**, **SequenceList** or **MemberList**, the **variableCharacteristics.valuesList** is not omitted or empty.
- For each component/variable, where **variableCharacteristics.dataType** is **OptionList**, **SequenceList** or **MemberList**, the **variableAttribute.value** is allowed based on the values in the provided **variableCharacteristics.valuesList**.
- For variables with **mutability** set to *WriteOnly* the variableAttribute.value is omitted in the **NotifyReportRequest**.
- For variables with **mutability** NOT set to *WriteOnly* the variableAttribute.value is NOT omitted in the **NotifyReportRequest**. There is one exception to this rule and that is for **EVSE.Power**. This variable only has a **maxLimit**.

TC_B_54_CS: Get Custom Report - with component/variable, but no instance

| | |
|-------------------|---|
| Test case name | Get Custom Report - with component/variable, but no instance |
| Test case Id | TC_B_54_CS |
| Use case Id(s) | B08 |
| Requirement(s) | B08.FR.01, B08.FR.03, B08.FR.04, B08.FR.11, B08.FR.12, B08.FR.14 |
| System under test | Charging Station |
| Description | CSMS requests a custom report for ItemsPerMessage of DeviceDataCtrlr. |
| Purpose | To test that Charging Station will send all instances if instance is not given. |
| Prerequisite(s) | Charging Station has implemented custom reporting (use case B08). |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State(s):

N/a

Main (Test scenario)

| Charging Station | CSMS |
|---|---|
| 2. Charging Station responds: GetReportResponse | 1. Test System sends GetReportRequest with: - requestId = <Generated requestId> - componentVariable.component.name = "DeviceDataCtrlr" - componentVariable.variable.name = "ItemsPerMessage" |
| 3. Charging Station sends NotifyReportRequest | 4. Test System responds with NotifyReportResponse |

Tool validations

* Step 2:

Message: **GetReportResponse** with:

- **status** = *Accepted*
- **statusInfo** is absent or **statusInfo** = "NoError"

* step 3:

Message: **NotifyReportRequest** with:

- **reportData[0].component.name** = "DeviceDataCtrlr"
- **reportData[0].variable.name** = "ItemsPerMessage"
- **reportData[0].variable.instance** = "GetReport"
- **reportData[1].component.name** = "DeviceDataCtrlr"
- **reportData[1].variable.name** = "ItemsPerMessage"
- **reportData[1].variable.instance** = "GetVariable"

Note: for AvailabilityState there will only be an *_Actual* value.It does not make any sense to have a *MinSet*, *MaxSet* or *Target* value for it.

Post scenario validations:

N/A

TC_B_55_CS: Get Custom Report - with component/variable/instance

| | |
|-------------------|---|
| Test case name | Get Custom Report - with component/variable/instance |
| Test case Id | TC_B_55_CS |
| Use case Id(s) | B08 |
| Requirement(s) | B08.FR.01, B08.FR.03, B08.FR.04, B08.FR.11, B08.FR.12, B08.FR.14 |
| System under test | Charging Station |
| Description | CSMS requests a custom report for ItemsPerMessage of DeviceDataCtrlr. |
| Purpose | To test that Charging Station will send one instances if instance is given. |
| Prerequisite(s) | Charging Station has implemented custom reporting (use case B08). |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State(s):

N/a

Main (Test scenario)

| Charging Station | CSMS |
|---|--|
| 2. Charging Station responds: GetReportResponse | 1. Test System sends GetReportRequest with: - requestId = <Generated requestId> - componentVariable.component.name = "DeviceDataCtrlr" - componentVariable.variable.name = "ItemsPerMessage" - componentVariable.instance = "GetReport" |
| 3. Charging Station sends NotifyReportRequest | 4. Test System responds with NotifyReportResponse |

Tool validations

* Step 2:

Message: **GetReportResponse** with:

- **status** = *Accepted*
- **statusInfo** is absent or **statusInfo** = "NoError"

* step 3:

Message: **NotifyReportRequest** with:

- **reportData[0].component.name** = "DeviceDataCtrlr"
- **reportData[0].variable.name** = "ItemsPerMessage"
- **reportData[0].variable.instance** = "GetReport"

Note: for AvailabilityState there will only be an *_Actual* value.It does not make any sense to have a *MinSet*, *MaxSet* or *Target* value for it._

Post scenario validations:

N/A

TC_B_56_CS: Get Custom Report - with component/variable, but no evseld

| | |
|-------------------|---|
| Test case name | Get Custom Report - with component/variable, but no evseld |
| Test case Id | TC_B_56_CS |
| Use case Id(s) | B08 |
| Requirement(s) | B08.FR.01, B08.FR.03, B08.FR.04, B08.FR.11, B08.FR.12, B08.FR.14 |
| System under test | Charging Station |
| Description | CSMS requests a custom report for AvailabilityState of EVSE |
| Purpose | To test that Charging Station will send all EVSEs when evseld is not given. |
| Prerequisite(s) | Charging Station has implemented custom reporting (use case B08). |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State(s):

N/a

Main (Test scenario)

| Charging Station | CSMS |
|---|--|
| 2. Charging Station responds: GetReportResponse | 1. Test System sends GetReportRequest with: - requestId = <Generated requestId> - componentVariable.component.name = "EVSE" - componentVariable.variable.name = "AvailabilityState" |
| 3. Charging Station sends NotifyReportRequest | 4. Test System responds with NotifyReportResponse |

Tool validations

* Step 2:

Message: **GetReportResponse** with:

- **status** = *Accepted*
- **statusInfo** is absent or **statusInfo** = "NoError"

* step 3:

Message: **NotifyReportRequest** with:

- **reportData[i].component.name** = "EVSE"
- **reportData[i].variable.name** = "AvailabilityState"
- **number of EVSEs** = <Configured EVSE count>

Note: for AvailabilityState there will only be an Actual value.

It does not make any sense to have a MinSet, MaxSet or Target value for it.

Post scenario validations:

N/A

TC_B_57_CS: Network Reconnection - After connection loss

| | |
|--------------------------|--|
| Test case name | Network Reconnection - After connection loss |
| Test case Id | TC_B_57_CS |
| Use case Id(s) | Part 4 section 5.3. Reconnecting |
| Requirement(s) | Described at section 5.3. |
| System under test | Charging Station |
| Description | When the connection is lost, the Charging Station SHALL try to reconnect. When reconnecting, the Charging Station SHALL use an increasing back-off time until it has successfully reconnected. |
| Purpose | To verify if the Charging Station is able to reconnect to the CSMS using the described OCPP reconnecting mechanism from part 4. |
| Prerequisite(s) | N/a |

Before (Preparations)

Configuration State:

OCPPCommCtrlr.NetworkProfileConnectionAttempts is 3

OCPPCommCtrlr.RetryBackOffRepeatTimes is 2

OCPPCommCtrlr.RetryBackOffRandomRange is 0

OCPPCommCtrlr.RetryBackOffWaitMinimum is <Configured RetryBackOffWaitMinimum>

Memory State:

N/a

Reusable State(s):

N/a

Main (Test scenario)

| Charging Station | CSMS |
|---|---|
| 1. The Test System closes the websocket connection. | |
| 2. The Charging Station waits for the duration of the configured RetryBackOffWaitMinimum and reconnects to the Test System. | 3. The connection attempt is accepted by the Test System. |
| 4. The Test System closes the websocket connection. | |
| 5. The Charging Station waits for the duration of the configured RetryBackOffWaitMinimum and reconnects to the Test System. | 6. The connection attempt is not accepted by the Test System. |
| 7. The Charging Station waits for double the reconnection time used at step 2 and reconnects to the Test System. | 8. The connection attempt is accepted by the Test System. |

Tool validations

* Step 2:

- The reconnection time is at least the configured RetryBackOffWaitMinimum.

* Step 7:

- The reconnection time is at least 2 times the reconnection time from step 2.

Post scenario validations:

- N/a

C Authorization

TC_C_02_CS: Local start transaction - Authorization Invalid/Unknown

| | |
|-------------------|--|
| Test case name | Local start transaction - Authorization Invalid/Unknown |
| Test case Id | TC_C_02_CS |
| Use case Id(s) | C01 OR C04 OR C06 |
| Requirement(s) | C01.FR.02 OR C06.FR.02 |
| System under test | Charging Station |
| Description | When a Charging Station needs to charge an EV, it needs to authorize the EV Driver first at the CSMS before the charging can be started or stopped. |
| Purpose | To verify whether the Charging Station is able to handle receiving an invalid idToken. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The Charging Station supports at least one authorization method described at the following Use cases; C01, C04, C06. - The Charging Station does NOT have a cable lock, which prevents the EV driver to connect the EV and EVSE before authorization. |

| |
|---|
| Before (Preparations) |
| Configuration State: AuthCtrlr.AuthEnabled is <i>true</i> (If implemented AND ReadWrite) AuthCtrlr.DisableRemoteAuthorization is <i>false</i> (If implemented) |
| Memory State: N/a |
| Reusable State(s): State is <i>EVConnectedPreSession</i> |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action:</u> Present idToken. | |
| 1. The Charging Station sends an AuthorizeRequest | 2. The Test System responds with an AuthorizeResponse with idTokenInfo.status <i>Invalid</i> |
| <u>Note(s):</u> - The Charging Station SHALL NOT send a TransactionEventRequest message after the AuthorizeRequest from step 1. - The Test System waits <Configured message timeout> seconds, before ending the testcase. | |

| |
|---|
| Tool validations |
| * Step 1: Message: AuthorizeRequest - idToken.idToken <Configured invalid_idtoken_idtoken> - idToken.type <Configured invalid_idtoken_type> |
| Post scenario validations: N/a |

TC_C_05_CS: Local start transaction - Authorization invalid - Cable lock

| | |
|-------------------|--|
| Test case name | Local start transaction - Authorization invalid - Cable lock |
| Test case Id | TC_C_05_CS |
| Use case Id(s) | C01 OR C04 OR C06 |
| Requirement(s) | C01.FR.02 OR C06.FR.02 |
| System under test | Charging Station |
| Description | When a Charging Station needs to charge an EV, it needs to authorize the EV Driver first before the charging can be started or stopped. |
| Purpose | To verify whether a Charging Station with a cable lock, which prevents the EV driver to connect the EV and EVSE before authorization, is able to handle receiving an invalid idToken. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The Charging Station has a cable lock, which prevents the EV driver to connect the EV and EVSE before authorization. - The Charging Station supports at least one authorization method described at the following Use cases; C01, C04, C06. - The Charging Station does NOT have the following configuration; TxStartPoint ReadOnly AND value Authorized is NOT set. |

| |
|---|
| Before (Preparations) |
| Configuration State: AuthCtrlr.AuthEnabled is <i>true</i> (If implemented AND ReadWrite) AuthCtrlr.DisableRemoteAuthorization is <i>false</i> (If implemented) |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action:</u> Present idToken. | |
| 1. The Charging Station sends an AuthorizeRequest | 2. The Test System responds with an AuthorizeResponse with idTokenInfo.status <i>Invalid</i> |
| <u>Note(s):</u> - The Charging Station SHALL NOT send a TransactionEventRequest message after the AuthorizeRequest from step 1. - The Test System waits <Configured message timeout> seconds, before ending the testcase. | |

| |
|--|
| Tool validations |
| * Step 1: Message: AuthorizeRequest - idToken.idToken <Configured invalid_idtoken_idtoken> - idToken.type <Configured invalid_idtoken_type> |
| Post scenario validations: N/a |

TC_C_04_CS: Local Stop Transaction - Different idToken

| | |
|-------------------|--|
| Test case name | Local Stop Transaction - Different idToken |
| Test case Id | TC_C_04_CS |
| Use case Id(s) | C01, C04, E07 |
| Requirement(s) | C01.FR.02, C01.FR.03 |
| System under test | Charging Station |
| Description | The EV Driver tries to stop an ongoing transaction, by locally presenting a different IdToken. |
| Purpose | To verify whether the Charging Station does not stop the charging session when a different idToken is presented, than the one used to start the transaction. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The Charging Station supports at least one authorization method described at the following Use cases; C01, C04. - The Charging Station does NOT use one idToken reader for multiple EVSE. - The Charging Station supports authorization methods other than NoAuthorization |

Before (Preparations)

Configuration State:

AuthCtrlr.AuthEnabled is *true* (If implemented AND ReadWrite)

AuthCtrlr.DisableRemoteAuthorization is *false* (If implemented)

Memory State:

- The "different idToken" does not exist in Authorization Cache or Local Authorization List.
- The "different idToken" does not have an associated GroupId that matches with the GroupId of the "starting idToken".

Reusable State(s):

State is *EnergyTransferStarted*

Main (Test scenario)

| | |
|---|---|
| Charging Station | CSMS |
| <u>Manual Action:</u> Present a different idToken than used to start the transaction. | |
| 1. The Charging Station sends an AuthorizeRequest | 2. The Test System responds with an AuthorizeResponse with idTokenInfo.status <i>Accepted</i> |
| <u>Note(s):</u> <ul style="list-style-type: none"> - The Charging Station SHALL NOT send a TransactionEventRequest message with an idToken field after receiving an idToken that is different, than the one used to start the transaction. - The Test System waits <Configured message timeout> seconds, before ending the testcase. | |

Tool validations

N/a

Post scenario validations:

- Charging Station has not sent a TransactionEventRequest(*Ended*).

TC_C_06_CS: Local start transaction - Authorization Blocked

| | |
|-------------------|--|
| Test case name | Local start transaction - Authorization Blocked |
| Test case Id | TC_C_06_CS |
| Use case Id(s) | C01 |
| Requirement(s) | C01.FR.02 |
| System under test | Charging Station |
| Description | When a Charging Station needs to charge an EV, it needs to authorize the EV Driver first at the CSMS before the charging can be started or stopped. |
| Purpose | To verify whether the Charging Station is able to handle receiving an Blocked idToken. |
| Prerequisite(s) | The Charging Station does NOT have a cable lock, which prevents the EV driver to connect the EV and EVSE before authorization. The Charging Station supports authorization methods other than NoAuthorization |

| |
|---|
| Before (Preparations) |
| Configuration State: AuthCtrlr.AuthEnabled is <i>true</i> (If implemented AND ReadWrite) AuthCtrlr.DisableRemoteAuthorization is <i>false</i> (If implemented) |
| Memory State: N/a |
| Reusable State(s): State is <i>EVConnectedPreSession</i> |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action</u> : Present idToken. | |
| 1. The Charging Station sends an AuthorizeRequest | 2. The Test System responds with an AuthorizeResponse with idTokenInfo.status <i>Blocked</i> |
| <u>Note(s)</u> : - The Charging Station <i>SHALL NOT</i> send a TransactionEventRequest message after the AuthorizeRequest from step 7. - The Test System waits <Configured message timeout> seconds, before ending the testcase. | |

| |
|---|
| Tool validations |
| * Step 1: Message: AuthorizeRequest - idToken.idToken <Configured blocked_idtoken_idtoken> - idToken.type <Configured blocked_idtoken_type> |
| Post scenario validations: N/a |

TC_C_07_CS: Local start transaction - Authorization Expired

| | |
|-------------------|--|
| Test case name | Local start transaction - Authorization Expired |
| Test case Id | TC_C_07_CS |
| Use case Id(s) | C01 |
| Requirement(s) | C01.FR.02 |
| System under test | Charging Station |
| Description | When a Charging Station needs to charge an EV, it needs to authorize the EV Driver first at the CSMS before the charging can be started or stopped. |
| Purpose | To verify whether the Charging Station is able to handle receiving an Expired idToken. |
| Prerequisite(s) | The Charging Station does NOT have a cable lock, which prevents the EV driver to connect the EV and EVSE before authorization. The Charging Station supports authorization methods other than NoAuthorization |

| |
|---|
| Before (Preparations) |
| Configuration State: AuthCtrlr.AuthEnabled is <i>true</i> (If implemented AND ReadWrite) AuthCtrlr.DisableRemoteAuthorization is <i>false</i> (If implemented) |
| Memory State: N/a |
| Reusable State(s): State is <i>EVConnectedPreSession</i> |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action</u> : Present idToken. | |
| 1. The Charging Station sends an AuthorizeRequest | 2. The Test System responds with an AuthorizeResponse with idTokenInfo.status <i>Expired</i> |
| <u>Note(s)</u> : - The Charging Station SHALL NOT send a TransactionEventRequest message after the AuthorizeRequest from step 7. - The Test System waits <Configured message timeout> seconds, before ending the testcase. | |

| |
|---|
| Tool validations |
| * Step 1: Message: AuthorizeRequest - idToken.idToken <Configured expired_idtoken_idtoken> - idToken.type <Configured expired_idtoken_type> |
| Post scenario validations: N/a |

TC_C_08_CS: Authorization through authorization cache - Accepted

| | |
|-------------------|---|
| Test case name | Authorization through authorization cache - Accepted |
| Test case Id | TC_C_08_CS |
| Use case Id(s) | C12 |
| Requirement(s) | C12_FR_02, C12_FR_04 |
| System under test | Charging Station |
| Description | This test case describes how the EV Driver is authorized to start a transaction while the Charging Station uses Cached IdToken. This enables the EV Driver to Online start a transaction by using the Authorization Cache in which the Charging Station can respond faster, as no AuthorizeRequest is being sent. |
| Purpose | To verify if the Charging Station is able to Authorize an idToken which has status "Accepted" in its cache according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | The Charging Station has an authorization cache AND the Charging Station supports at least one of the following local start authorization options: C-30, C-31, C-32, C-34 OR supports at least one of the following remote start authorization options: C-36, C-37 |

| |
|---|
| Before (Preparations) |
| Configuration State: AuthCacheCtrlr.AuthCacheEnabled is <i>true</i> (If implemented) AuthCtrlr.LocalPreAuthorize is <i>true</i> (If implemented) |
| Memory State: <i>IdTokenCached</i> for <Configured valid IdToken fields> |
| Reusable State(s): N/a |

| | |
|--|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>Authorized</i> (Cached idToken) | |
| 2. Execute Reusable State <i>EVConnectedPreSession</i> | |
| 3. Execute Reusable State <i>EnergyTransferStarted</i> | |

| |
|---|
| Tool validations |
| N/a |
| Post scenario validations: - Energy transfer is started |

NOTE | If the Charging Station supports ISO15118, this testcase needs to be executed using EIM.

TC_C_09_CS: Authorization through authorization cache - Invalid & Not Accepted

| | |
|-------------------|---|
| Test case name | Authorization through authorization cache - Invalid & Not Accepted |
| Test case Id | TC_C_09_CS |
| Use case Id(s) | C12 |
| Requirement(s) | C12_FR_05, C10_FR_03 |
| System under test | Charging Station |
| Description | This test case describes how the EV Driver is authorized to start a transaction while the Charging Station uses Cached IdToken. This enables the EV Driver to Online start a transaction by using the Authorization Cache in which the Charging Station can respond faster, as no AuthorizeRequest is being sent. |
| Purpose | To verify if the Charging Station is able to Authorize an idToken which has status "Invalid" in its cache according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | The Charging Station has an authorization cache AND the Charging Station supports at least one of the following local start authorization options: C-30, C-31, C-32, C-34 OR supports at least one of the following remote start authorization options: C-36, C-37 |

| |
|--|
| Before (Preparations) |
| Configuration State: AuthCacheCtrlr.AuthCacheEnabled is <i>true</i> (If implemented) AuthCtrlr.LocalPreAuthorize is <i>true</i> (If implemented) AuthCacheCtrlr.DisablePostAuthorize is <i>false</i> (If implemented) |
| Memory State: <i>IdTokenCached</i> for <Configured invalid IdToken fields> |
| Reusable State(s): N/a |

| | |
|--|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>Authorized</i> for <Configured invalid IdToken fields> | |

| |
|--|
| Tool validations |
| N/a |
| Post scenario validations: - N/a |

NOTE | If the Charging Station supports ISO15118, this testcase needs to be executed using EIM.

TC_C_10_CS: Authorization through authorization cache - Blocked

| | |
|-------------------|---|
| Test case name | Authorization through authorization cache - Blocked |
| Test case Id | TC_C_10_CS |
| Use case Id(s) | C12 |
| Requirement(s) | C12_FR_05, C10_FR_03 |
| System under test | Charging Station |
| Description | This test case describes how the EV Driver is authorized to start a transaction while the Charging Station uses Cached IdToken. This enables the EV Driver to Online start a transaction by using the Authorization Cache in which the Charging Station can respond faster, as no AuthorizeRequest is being sent. |
| Purpose | To verify if the Charging Station is able to Authorize an idToken which has status "Blocked" in its cache according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | The Charging Station has an authorization cache AND the Charging Station supports at least one of the following local start authorization options: C-30, C-31, C-32, C-34 OR supports at least one of the following remote start authorization options: C-36, C-37 |

Before (Preparations)

Configuration State:

AuthCacheCtrlr.AuthCacheEnabled is *true* (If implemented)

AuthCtrlr.LocalPreAuthorize is *true* (If implemented)

AuthCacheCtrlr.DisablePostAuthorize is *false* (If implemented)

Memory State:

IdTokenCached for <Configured blocked IdToken fields>

Reusable State(s):

N/a

Main (Test scenario)

| | |
|------------------|------|
| Charging Station | CSMS |
|------------------|------|

1. Execute **Reusable State** *Authorized* for <Configured blocked IdToken fields>

Tool validations

N/a

Post scenario validations:

- N/a

NOTE | If the Charging Station supports ISO15118, this testcase needs to be executed using EIM.

TC_C_11_CS: Authorization through authorization cache - Expired

| | |
|-------------------|---|
| Test case name | Authorization through authorization cache - Expired |
| Test case Id | TC_C_11_CS |
| Use case Id(s) | C12 |
| Requirement(s) | C12_FR_05, C10_FR_03 |
| System under test | Charging Station |
| Description | This test case describes how the EV Driver is authorized to start a transaction while the Charging Station uses Cached IdToken. This enables the EV Driver to Online start a transaction by using the Authorization Cache in which the Charging Station can respond faster, as no AuthorizeRequest is being sent. |
| Purpose | To verify if the Charging Station is able to Authorize an idToken which has status "Expired" in its cache according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | The Charging Station has an authorization cache AND the Charging Station supports at least one of the following local start authorization options: C-30, C-31, C-32, C-34 OR supports at least one of the following remote start authorization options: C-36, C-37 |

| |
|--|
| Before (Preparations) |
| Configuration State: AuthCacheCtrlr.AuthCacheEnabled is <i>true</i> (If implemented) AuthCtrlr.LocalPreAuthorize is <i>true</i> (If implemented) AuthCacheCtrlr.DisablePostAuthorize is <i>false</i> (If implemented) |
| Memory State: <i>IdTokenCached</i> for <Configured expired IdToken fields> |
| Reusable State(s): N/a |

| | |
|--|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>Authorized</i> for <Configured expired IdToken fields> | |

| |
|--|
| Tool validations |
| N/a |
| Post scenario validations: - N/a |

NOTE | If the Charging Station supports ISO15118, this testcase needs to be executed using EIM.

TC_C_12_CS: Authorization through authorization cache - Invalid & Accepted

| | |
|-------------------|---|
| Test case name | Authorization through authorization cache - Invalid & Accepted |
| Test case Id | TC_C_12_CS |
| Use case Id(s) | C12 |
| Requirement(s) | C12_FR_05, C10_FR_03 |
| System under test | Charging Station |
| Description | This test case describes how the EV Driver is authorized to start a transaction while the Charging Station uses Cached IdToken. This enables the EV Driver to Online start a transaction by using the Authorization Cache in which the Charging Station can respond faster, as no AuthorizeRequest is being sent. |
| Purpose | To verify if the Charging Station is able to Authorize an idToken which has status "Invalid" in its cache, but the CSMS has status "Valid", according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | The Charging Station has an authorization cache AND the Charging Station supports at least one of the following local start authorization options: C-30, C-31, C-32, C-34 OR supports at least one of the following remote start authorization options: C-36, C-37 |

| |
|---|
| Before (Preparations) |
| Configuration State: AuthCacheCtrlr.AuthCacheEnabled is <i>true</i> (If implemented) AuthCtrlr.LocalPreAuthorize is <i>true</i> (If implemented) AuthCacheCtrlr.DisablePostAuthorize is <i>false</i> (If implemented) |
| Memory State: IdTokenCached for <Configured invalid IdToken fields> |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|---|------|
| Charging Station | CSMS |
| <p>1. Execute Reusable State <i>Authorized</i> for <Configured invalid IdToken fields> (idTokenInfo.status Accepted) <u>Note:</u> <i>The configured invalid idToken fields are being used, however the Test system responds with status Accepted to simulate an outdated cache entry. In the mean time the idToken became valid.</i></p> | |
| <p>2. Execute Reusable State <i>EVConnectedPreSession</i></p> | |
| <p>3. Execute Reusable State <i>EnergyTransferStarted</i></p> | |

| |
|---|
| Tool validations |
| N/a |
| Post scenario validations: - Energy transfer is started |

NOTE | If the Charging Station supports ISO15118, this testcase needs to be executed using EIM.

TC_C_13_CS: Authorization through authorization cache - Accepted but cable not connected yet.

| | |
|-------------------|---|
| Test case name | Authorization through authorization cache - Accepted but cable not connected yet. |
| Test case Id | TC_C_13_CS |
| Use case Id(s) | C12 |
| Requirement(s) | C12_FR_02, C12_FR_04 |
| System under test | Charging Station |
| Description | This test case describes how the EV Driver is authorized to start a transaction while the Charging Station uses Cached IdToken. This enables the EV Driver to Online start a transaction by using the Authorization Cache in which the Charging Station can respond faster, as no AuthorizeRequest is being sent. |
| Purpose | To verify if the Charging Station is able to Authorize an idToken which has status "Accepted" in its cache but the cable is not connected yet according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | The Charging Station has an authorization cache AND the Charging Station supports at least one of the following local start authorization options: C-30, C-31, C-32, C-34 OR supports at least one of the following remote start authorization options: C-36, C-37 |

| |
|---|
| Before (Preparations) |
| Configuration State: AuthCacheCtrlr.AuthCacheEnabled is <i>true</i> (If implemented) AuthCtrlr.LocalPreAuthorize is <i>true</i> (If implemented) |
| Memory State: <i>IdTokenCached</i> for <Configured valid IdToken fields> |
| Reusable State(s): N/a |

| | |
|--|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>Authorized</i> (Cached idToken) | |
| 2. Execute Reusable State <i>EVConnectedPreSession</i> | |
| 3. Execute Reusable State <i>EnergyTransferStarted</i> | |

| |
|---|
| Tool validations |
| N/a |
| Post scenario validations: - Energy transfer is started |

NOTE | If the Charging Station supports ISO15118, this testcase needs to be executed using EIM.

TC_C_14_CS: Authorization through authorization cache - GroupID equal to MasterPassGroupId.

| | |
|-------------------|---|
| Test case name | Authorization through authorization cache - GroupID equal to MasterPassGroupId. |
| Test case Id | TC_C_14_CS |
| Use case Id(s) | C12 |
| Requirement(s) | C12.FR.09, C16.FR.03 |
| System under test | Charging Station |
| Description | This test case describes how the EV Driver is authorized to start a transaction while the Charging Station uses Cached IdToken. This enables the EV Driver to Online start a transaction by using the Authorization Cache in which the Charging Station can respond faster, as no AuthorizeRequest is being sent. |
| Purpose | To verify if the Charging Station is able to respond correctly to an idToken which has the "MasterPassGroupId" as group id according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | AuthCacheCtrlr.Available is implemented with value true The Charging station supports MasterPass feature. The Charging Station supports authorization methods other than NoAuthorization |

| |
|---|
| Before (Preparations) |
| Configuration State: AuthCacheCtrlr.AuthCacheEnabled is <i>true</i> (If implemented) AuthCtrlr.LocalPreAuthorize is <i>true</i> (If implemented) AuthCtrlr.MasterPassGroupId is <i><Configured MasterPassGroupId></i> |
| Memory State: Store the idToken that is part of the MasterPass group at the cache. |
| Reusable State(s): State is <i>EnergyTransferStarted</i> |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action</u> : Present the idToken with group id "MasterPassGroupId" which is already configured in the Authorization Cache | |
| 1. The Charging Station sends a TransactionEventRequest | 2. The Test System responds with a TransactionEventResponse |
| 3. Execute Reusable State <i>EVConnectedPostSession</i> | |
| 4. Execute Reusable State <i>EVDisconnected</i> | |
| 5. Execute Reusable State <i>ParkingBayUnoccupied</i> | |
| <u>Note(s)</u> : Step 3, 4, and 5 are only executed if the transaction is still running. | |

| |
|---|
| Tool validations |
| * Step 2: Message TransactionEventRequest - triggerReason <i>StopAuthorized</i> - idToken.idToken <i><Configured masterPass idToken></i> - idToken.type <i><Configured masterPass idTokenType></i> If eventType <i>Ended</i> , then: - transactionInfo.stoppedReason <i>MasterPass</i> |
| Post scenario validations: - N/a |

NOTE | If the Charging Station supports ISO15118, this testcase needs to be executed using EIM.

TC_C_15_CS: Authorization through authorization cache - StopTxOnInvalidId = false, MaxEnergyOnInvalidId > 0

| | |
|-------------------|--|
| Test case name | Authorization through authorization cache - StopTxOnInvalidId = false, MaxEnergyOnInvalidId > 0 |
| Test case Id | TC_C_15_CS |
| Use case Id(s) | C12 |
| Requirement(s) | C12_FR_02, C12_FR_04 |
| System under test | Charging Station |
| Description | This test case describes how the EV Driver is authorized to start a transaction while the Charging Station uses Cached IdToken. This enables the EV Driver to Online start a transaction by using the Authorization Cache in which the Charging Station can respond faster, as no AuthorizeRequest is being sent. |
| Purpose | To verify if the Charging Station is able to respond correctly to an idToken which has status "Accepted" in its cache but is "Invalid" in the CSMS with certain values of StopTxOnInvalidId and MaxEnergyOnInvalidId according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | <ul style="list-style-type: none"> - AuthCacheCtrlr.Available is implemented with value true - The Charging Station has MaxEnergyOnInvalidId implemented - At least one of the following must be supported; Local auth list, auth cache, StartTxUnknownIds. - The Charging Station supports authorization methods other than NoAuthorization |

| |
|--|
| Before (Preparations) |
| <p>Configuration State:</p> <p>AuthCacheCtrlr.AuthCacheEnabled is <i>true</i> (If implemented)</p> <p>AuthCtrlr.LocalPreAuthorize is <i>true</i> (If implemented)</p> <p>AuthCtrlr.LocalAuthorizeOffline is <i>true</i></p> <p>OfflineTxForUnknownIdEnabled is <i>true</i> (If implemented)</p> <p>StopTxOnInvalidId is <i>false</i></p> <p>MaxEnergyOnInvalidId is <i>10.000</i></p> <p>OfflineThreshold is <i><Configured RetryBackOffWaitMinimum_duration> + 60.0</i></p> <p>RetryBackOffWaitMinimum is <i><Configured RetryBackOffWaitMinimum_duration></i></p> <p>RetryBackOffRandomRange is <i>0</i></p> <p><u>Note:</u></p> <p><i><Configured RetryBackOffWaitMinimum_duration should be long enough to execute manual tasks></i></p> <p>Memory State:</p> <p><i>IdTokenCached</i> for <i><Configured valid IdToken fields></i> (If implemented) A known valid idToken is configured in the Local auth list (if implemented) and auth cache (if implemented)</p> <p>Reusable State(s):</p> <p>State is <i>EVConnectedPreSession</i></p> |

| Main (Test scenario) | |
|---|---|
| Charging Station | CSMS |
| 1. The Test System closes the WebSocket connection AND does not accept a reconnect. | |
| <u>Manual Action</u> : Present valid idToken which is already configured in the Authorization Cache | |
| <u>Note(s)</u> : The Test System will wait for _<Configured Transaction Duration> seconds_ | |
| 2. The Test System accepts reconnection attempt from the Charging Station. | |
| <u>Note(s)</u> : - The Charging Station will empty its Transaction message queue. This will contain one or more TransactionEventRequest messages | |
| 3. The Charging Station sends a TransactionEventRequest | 4. The Test System responds with a TransactionEventResponse with idTokenInfo.status <i>Invalid</i> (if idToken is not omitted) |

| Tool validations |
|--|
| <p>* Step 3:</p> <p>Message TransactionEventRequest</p> <p>A message with (optional):</p> <ul style="list-style-type: none">- triggerReason <i>Authorized</i>- idToken.idToken <i><Configured valid_idtoken_idtoken></i>- idToken.type <i><Configured valid_idtoken_type></i>- offline <i>True</i> <p>A message with:</p> <ul style="list-style-type: none">- triggerReason <i>ChargingStateChanged</i>- offline <i>True</i> <p>No message with:</p> <ul style="list-style-type: none">- triggerReason <i>Deauthorized</i> or- triggerReason <i>ChargingStateChanged</i> and- transactionInfo.chargingState <i>SuspendedEVSE</i> |
| <p>Post scenario validations:</p> <ul style="list-style-type: none">- Energy transfer is started but only MaxEnergyOnInvalidId amount of energy is delivered |

NOTE

If the Charging Station supports ISO15118, this testcase needs to be executed using EIM.

TC_C_16_CS: Authorization through authorization cache - StopTxOnInvalidId = true

| | |
|-------------------|---|
| Test case name | Authorization through authorization cache - StopTxOnInvalidId = true |
| Test case Id | TC_C_16_CS |
| Use case Id(s) | C12 |
| Requirement(s) | C12.FR.02, C12.FR.04, C15.FR.04 |
| System under test | Charging Station |
| Description | This test case describes how the EV Driver is authorized to start a transaction while the Charging Station uses Cached IdToken. This enables the EV Driver to Online start a transaction by using the Authorization Cache in which the Charging Station can respond faster, as no AuthorizeRequest is being sent. |
| Purpose | To verify if the Charging Station is able to respond correctly to an idToken which has status "Accepted" in its cache but is "Invalid" in the CSMS when StopTxOnInvalidId is true according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | <ul style="list-style-type: none"> - AuthCacheCtrlr.Available is implemented with value true - At least one of the following must be supported; Local auth list, auth cache, StartTxUnknownIds. - The Charging Station supports authorization methods other than NoAuthorization |

| |
|--|
| Before (Preparations) |
| Configuration State: AuthCacheEnabled is <i>true</i> (If implemented) LocalPreAuthorize is <i>true</i> (If implemented) LocalAuthorizeOffline is <i>true</i> StopTxOnInvalidId is <i>true</i> MaxEnergyOnInvalidId is <i>0</i> |
| Memory State: IdTokenCached for <Configured valid IdToken fields> (If implemented) A known valid idToken is configured in the Local auth list (if implemented) and auth cache (if implemented) |
| Reusable State(s): State is <i>EVConnectedPreSession</i> |

| Main (Test scenario) | |
|---|---|
| Charging Station | CSMS |
| 1. The Test System closes the WebSocket connection AND does not accept a reconnect. | |
| <u>Manual Action</u> : Present valid idToken which is already configured in the Authorization Cache | |
| <u>Note(s)</u> : The Test System will wait for 5 seconds | |
| 2. The Test System accepts reconnection attempt from the Charging Station. | |
| <u>Note(s)</u> : - The Charging Station will empty its Transaction message queue. This will contain one or more TransactionEventRequest messages | |
| 3. The Charging Station sends a TransactionEventRequest | 4. The Test System responds with a TransactionEventResponse with idTokenInfo.status Invalid (if idToken is not omitted) |
| 5. The Charging Station sends a TransactionEventRequest with triggerReason Deauthorized | 6. The Test System responds with a TransactionEventResponse |

| Tool validations |
|---|
| <p>* Step 3:</p> <p>Message TransactionEventRequest</p> <p>A message with (optional):</p> <ul style="list-style-type: none">- triggerReason <i>Authorized</i>- idToken.idToken <i><Configured valid_idtoken_idtoken></i>- idToken.type <i><Configured valid_idtoken_type></i>- offline <i>True</i> <p>A message with:</p> <ul style="list-style-type: none">- triggerReason <i>ChargingStateChanged</i>- offline <i>True</i> <p>A message with:</p> <ul style="list-style-type: none">- triggerReason <i>Deauthorized</i> |
| <p>Post scenario validations:</p> <ul style="list-style-type: none">- Energyflow stops on receiving status invalid |

NOTE

|

If the Charging Station supports ISO15118, this testcase needs to be executed using EIM.

TC_C_17_CS: Authorization through authorization cache - StopTxOnInvalidId = false

| | |
|-------------------|---|
| Test case name | Authorization through authorization cache - StopTxOnInvalidId = false |
| Test case Id | TC_C_17_CS |
| Use case Id(s) | C12 |
| Requirement(s) | C12.FR.02, C12.FR.04. C15.FR.06 |
| System under test | Charging Station |
| Description | This test case describes how the EV Driver is authorized to start a transaction while the Charging Station uses Cached IdToken. This enables the EV Driver to Online start a transaction by using the Authorization Cache in which the Charging Station can respond faster, as no AuthorizeRequest is being sent. |
| Purpose | To verify if the Charging Station is able to respond correctly to an idToken which has status "Accepted" in its cache but is "Invalid" in the CSMS when StopTxOnInvalidId is false according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | <ul style="list-style-type: none"> - AuthCacheCtrlr.Available is implemented with value true - At least one of the following must be supported; Local auth list, auth cache, StartTxUnknownIds. - The Charging Station supports authorization methods other than NoAuthorization |

| |
|---|
| Before (Preparations) |
| Configuration State: AuthCacheEnabled is <i>true</i> (If implemented) LocalPreAuthorize is <i>true</i> (If implemented) OfflineTxForUnknownIdEnabled is <i>true</i> (If implemented) StopTxOnInvalidId is <i>false</i> MaxEnergyOnInvalidId is <i>0</i> |
| Memory State: IdTokenCached for <Configured valid IdToken fields> (If implemented) A known valid idToken is configured in the Local auth list (if implemented) and auth cache (if implemented) |
| Reusable State(s): State is <i>EVConnectedPreSession</i> |

| Main (Test scenario) | |
|---|---|
| Charging Station | CSMS |
| 1. The Test System closes the WebSocket connection AND does not accept a reconnect. | |
| <u>Manual Action</u> : Present valid idToken which is already configured in the Authorization Cache | |
| <u>Note(s)</u> : The Test System will wait for 5 seconds | |
| 2. The Test System accepts reconnection attempt from the Charging Station. | |
| <u>Note(s)</u> : - The Charging Station will empty its Transaction message queue. This will contain one or more TransactionEventRequest messages | |
| 3. The Charging Station sends a TransactionEventRequest | 4. The Test System responds with a TransactionEventResponse with idTokenInfo.status <i>Invalid</i> (if idToken is not omitted) |
| 5. The Charging Station sends a TransactionEventRequest with triggerReason <i>SuspendedEVSE</i> | 6. The Test System responds with a TransactionEventResponse |

| Tool validations |
|--|
| <p>* Step 2:</p> <p>Message TransactionEventRequest</p> <p>A message with:</p> <ul style="list-style-type: none">- triggerReason <i>Authorized</i>- idToken.idToken <i><Configured valid_idtoken_idtoken></i>- idToken.type <i><Configured valid_idtoken_type></i>- offline <i>True</i> <p>A message with:</p> <ul style="list-style-type: none">- transactionInfo.chargingState <i>SuspendedEVSE</i> <p>No message with: - triggerReason <i>SuspendedEVSE</i></p> |
| <p>Post scenario validations:</p> <ul style="list-style-type: none">- Energyflow stops on receiving status invalid |

NOTE

If the Charging Station supports ISO15118, this testcase needs to be executed using EIM.

TC_C_18_CS: Authorization through authorization cache - StopTxOnInvalidId = true, MaxEnergyOnInvalidId > 0

| | |
|-------------------|---|
| Test case name | Authorization through authorization cache - StopTxOnInvalidId = true, MaxEnergyOnInvalidId > 0 |
| Test case Id | TC_C_18_CS |
| Use case Id(s) | C12 |
| Requirement(s) | C12_FR_02, C12_FR_04 |
| System under test | Charging Station |
| Description | This test case describes how the EV Driver is authorized to start a transaction while the Charging Station uses Cached IdToken. This enables the EV Driver to Online start a transaction by using the Authorization Cache in which the Charging Station can respond faster, as no AuthorizeRequest is being sent. |
| Purpose | To verify if the Charging Station is able to respond correctly to an idToken which has status "Accepted" in its cache but is "Invalid" in the CSMS when StopTxOnInvalidId is true and MaxEnergyOnInvalidId > 0 according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | <ul style="list-style-type: none"> - AuthCacheCtrlr.Available is implemented with value true - The Charging Station has MaxEnergyOnInvalidId implemented. - At least one of the following must be supported; Local auth list, auth cache, StartTxUnknownIds. - The Charging Station supports authorization methods other than NoAuthorization |

| |
|--|
| Before (Preparations) |
| <p>Configuration State:</p> <p>AuthCacheEnabled is true (If implemented)</p> <p>LocalPreAuthorize is true (If implemented)</p> <p>LocalAuthorizeOffline is true</p> <p>OfflineTxForUnknownIdEnabled is true (If implemented)</p> <p>StopTxOnInvalidId is true</p> <p>MaxEnergyOnInvalidId is 500</p> <p>OfflineThreshold is <Configured RetryBackOffWaitMinimum_duration> + 60.0</p> <p>RetryBackOffWaitMinimum is <Configured RetryBackOffWaitMinimum_duration></p> <p>RetryBackOffRandomRange is 0</p> <p><u>Note:</u></p> <p><Configured RetryBackOffWaitMinimum_duration should be long enough to execute manual tasks></p> <p>Memory State:</p> <p>IdTokenCached for <Configured valid IdToken fields> (If implemented)</p> <p>A known valid idToken is configured in the Local auth list (if implemented) and auth cache (if implemented)</p> <p>Reusable State(s):</p> <p>State is EVConnectedPreSession</p> |

| Main (Test scenario) | |
|---|---|
| Charging Station | CSMS |
| 1. The Test System closes the WebSocket connection AND does not accept a reconnect. | |
| <u>Manual Action</u> : Present valid idToken which is already configured in the Authorization Cache | |
| <u>Note(s)</u> : The Test System will wait for _<Configured Transaction Duration> seconds_ | |
| 2. The Test System accepts reconnection attempt from the Charging Station. | |
| <u>Note(s)</u> : - The Charging Station will empty its Transaction message queue. This will contain one or more TransactionEventRequest messages | |
| 3. The Charging Station sends a TransactionEventRequest | 4. The Test System responds with a TransactionEventResponse with idTokenInfo.status Invalid (if idToken is not omitted) |
| 5. The Charging Station sends a TransactionEventRequest with triggerReason Deauthorized | 6. The Test System responds with a TransactionEventResponse |

| Tool validations |
|--|
| <p>* Step 3:</p> <p>Message TransactionEventRequest</p> <p>A message with (optional):</p> <ul style="list-style-type: none">- triggerReason <i>Authorized</i>- idToken.idToken <i><Configured valid_idtoken_idtoken></i>- idToken.type <i><Configured valid_idtoken_type></i>- offline <i>True</i> <p>A message with:</p> <ul style="list-style-type: none">- triggerReason <i>ChargingStateChanged</i>- offline <i>True</i> <p>* Step 5:</p> <p>A message with:</p> <ul style="list-style-type: none">- triggerReason <i>Deauthorized</i>- offline <i>False</i> |
| <p>Post scenario validations:</p> <ul style="list-style-type: none">- Energyflow stops on receiving status invalid |

NOTE

If the Charging Station supports ISO15118, this testcase needs to be executed using EIM.

TC_C_57_CS: Authorization through authorization cache - AuthCacheDisablePostAuthorize

| | |
|-------------------|---|
| Test case name | Authorization through authorization cache - AuthCacheDisablePostAuthorize |
| Test case Id | TC_C_57_CS |
| Use case Id(s) | C12 |
| Requirement(s) | C12.FR.05, C10.FR.03 |
| System under test | Charging Station |
| Description | This test case describes how the EV Driver can be authorized to start a transaction by using Cached IdTokens. This enables the EV Driver to start a transaction while the Charging Station is online by using the Authorization Cache in which case the Charging Station can respond faster, since no AuthorizeRequest is being sent. For an IdToken that does not have status "Accepted" the Charging Station will still send an AuthorizeRequest to check against the most recent status from CSMS. However, when the setting AuthCacheDisablePostAuthorize is set to true, then the Charging Station will not do this. |
| Purpose | To verify that the Charging Station will not send an AuthorizeRequest for an IdToken in the Authorization Cache that is not "Accepted", when AuthCacheDisablePostAuthorize is set to true. |
| Prerequisite(s) | AuthCacheCtrlr.DisablePostAuthorize is implemented AND The Charging Station has an authorization cache AND the Charging Station supports at least one of the following local start authorization options: C-30, C-31, C-32, C-34 OR supports at least one of the following remote start authorization options: C-36, C-37 |

| |
|---|
| Before (Preparations) |
| Configuration State: AuthCacheCtrlr.Enabled is true (If implemented) AuthCtrlr.LocalPreAuthorize is true (If implemented) AuthCacheCtrlr.DisablePostAuthorize is true |
| Memory State: IdTokenCached for <Configured invalid IdToken fields> |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action</u> : Present Invalid idToken which is already configured in the Authorization Cache | |
| <u>Note</u> : This step is executed if configured scenario is Local | |
| 2. The Charging Station responds with a RequestStartTransactionResponse | 1. The Test System sends a RequestStartTransactionRequest with idToken.idToken <Configured invalid_idtoken_idtoken> idToken.type <Configured invalid_idtoken_type> evseld <Configured evseld> <u>Note</u> : This step is executed if configured scenario is Remote |
| 3. The Charging Station does NOT send a AuthorizeRequest | |
| 4. Execute Reusable State <i>EVConnectedPreSession</i> | |
| 5. The Charging Station does NOT start charging. | |

| |
|--|
| Tool validations |
| * Step 3: Check that Charging Station does NOT send an AuthorizeRequest and authorization is refused. |
| Post scenario validations: The Charging Station does NOT start charging. |

NOTE

If the Charging Station supports ISO15118, this testcase needs to be executed using EIM.

TC_C_27_CS: Online authorization through local authorization list - Accepted

| | |
|-------------------|---|
| Test case name | Online authorization through local authorization list - Accepted |
| Test case Id | TC_C_27_CS |
| Use case Id(s) | C14 |
| Requirement(s) | C14_FR_02 |
| System under test | Charging Station |
| Description | This test case describes how to authorize an IdToken via the Local Authorization List while the Charging Station is online. When online the Charging Station can then locally authorize the IdToken, and is not required to send an AuthorizeRequest for a known IdToken. |
| Purpose | To verify if the Charging Station is able to authorize an idToken which has status "Accepted" in its local authorization list according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | - LocalAuthListCtrlr.LocalAuthListAvailable is implemented with value true AND - The Charging Station must support an authorization method other than NoAuthorization or Central |

| |
|--|
| Before (Preparations) |
| Configuration State: LocalAuthListEnabled is <i>true</i> (If implemented) LocalPreAuthorize is <i>true</i> (If implemented) |
| Memory State: A known valid idToken is configured in the Local Authorization List |
| Reusable State(s): N/a |

| | |
|--|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>Authorized</i> (Stored idToken) | |
| 2. Execute Reusable State <i>EVConnectedPreSession</i> | |
| 3. Execute Reusable State <i>EnergyTransferStarted</i> | |

| |
|--|
| Tool validations |
| N/a |
| Post scenario validations: - Energy is transferred |

TC_C_28_CS: Online authorization through local authorization list - Invalid & Not Accepted

| | |
|-------------------|---|
| Test case name | Online authorization through local authorization list - Invalid & Not Accepted |
| Test case Id | TC_C_28_CS |
| Use case Id(s) | C14 |
| Requirement(s) | C14_FR_03 |
| System under test | Charging Station |
| Description | This test case describes how to authorize an IdToken via the Local Authorization List while the Charging Station is online. When online the Charging Station can then locally authorize the IdToken, and is not required to send an AuthorizeRequest for a known IdToken. |
| Purpose | To verify if the Charging Station is able to respond correctly to an idToken which has status "Invalid" in its local authorization list according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | - LocalAuthListCtrlr.LocalAuthListAvailable is implemented with value true AND - The Charging Station must support an authorization method other than NoAuthorization or Central |

| |
|---|
| Before (Preparations) |
| Configuration State: LocalAuthListEnabled is <i>true</i> (If implemented) LocalPreAuthorize is <i>true</i> (If implemented) LocalAuthListDisablePostAuthorize <i>false</i> (If implemented) |
| Memory State: A known invalid idToken is configured in the Local Authorization List |
| Reusable State(s): N/a |

| | |
|---|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>Authorized</i> (idTokenInfo.status Invalid) | |

| |
|--|
| Tool validations |
| N/a |
| Post scenario validations: - N/a |

TC_C_29_CS: Online authorization through local authorization list - Blocked

| | |
|-------------------|---|
| Test case name | Online authorization through local authorization list - Blocked |
| Test case Id | TC_C_29_CS |
| Use case Id(s) | C14 |
| Requirement(s) | C14_FR_03 |
| System under test | Charging Station |
| Description | This test case describes how to authorize an IdToken via the Local Authorization List while the Charging Station is online. When online the Charging Station can then locally authorize the IdToken, and is not required to send an AuthorizeRequest for a known IdToken. |
| Purpose | To verify if the Charging Station is able to respond correctly to an idToken which has status "Blocked" in its local authorization list according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | - LocalAuthListCtrlr.LocalAuthListAvailable is implemented with value true AND - The Charging Station must support an authorization method other than NoAuthorization or Central |

| |
|--|
| Before (Preparations) |
| Configuration State: LocalAuthListEnabled is <i>true</i> (If implemented) LocalPreAuthorize is <i>true</i> (If implemented) <i>*LocalAuthListDisablePostAuthorize * false</i> (If implemented) |
| Memory State: A known blocked idToken is configured in the Local Authorization List |
| Reusable State(s): N/a |

| | |
|---|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>Authorized</i> (idTokenInfo.status Blocked) | |

| |
|--|
| Tool validations |
| N/a |
| Post scenario validations: - N/a |

TC_C_30_CS: Online authorization through local authorization list - Expired

| | |
|--------------------------|---|
| Test case name | Online authorization through local authorization list - Expired |
| Test case Id | TC_C_30_CS |
| Use case Id(s) | C14 |
| Requirement(s) | C14_FR_03 |
| System under test | Charging Station |
| Description | This test case describes how to authorize an IdToken via the Local Authorization List while the Charging Station is online. When online the Charging Station can then locally authorize the IdToken, and is not required to send an AuthorizeRequest for a known IdToken. |
| Purpose | To verify if the Charging Station is able to respond correctly to an idToken which has status "Expired" in its local authorization list according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | <ul style="list-style-type: none"> - LocalAuthListCtrlr.LocalAuthListAvailable is implemented with value true AND - The Charging Station must support an authorization method other than NoAuthorization or Central |

| |
|--|
| Before (Preparations) |
| Configuration State: LocalAuthListEnabled is <i>true</i> (If implemented) LocalPreAuthorize is <i>true</i> (If implemented) <i>*LocalAuthListDisablePostAuthorize * false</i> (If implemented) |
| Memory State: A known expired idToken is configured in the Local Authorization List |
| Reusable State(s): N/a |

| | |
|---|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>Authorized</i> (idTokenInfo.status Expired) | |

| |
|--|
| Tool validations |
| N/a |
| Post scenario validations: - N/a |

TC_C_31_CS: Online authorization through local authorization list - Invalid & Accepted

| | |
|-------------------|---|
| Test case name | Online authorization through local authorization list - Invalid & Accepted |
| Test case Id | TC_C_31_CS |
| Use case Id(s) | C14 |
| Requirement(s) | C14_FR_03 |
| System under test | Charging Station |
| Description | This test case describes how to authorize an IdToken via the Local Authorization List while the Charging Station is online. When online the Charging Station can then locally authorize the IdToken, and is not required to send an AuthorizeRequest for a known IdToken. |
| Purpose | To verify if the Charging Station is able to respond correctly to an idToken which has status "Invalid" in its local authorization list, but is actually valid for the CSMS, according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | - LocalAuthListCtrlr.LocalAuthListAvailable is implemented with value true AND - The Charging Station must support an authorization method other than NoAuthorization or Central |

| |
|--|
| Before (Preparations) |
| Configuration State: LocalAuthListEnabled is <i>true</i> (If implemented) LocalPreAuthorize is <i>true</i> (If implemented) <i>*LocalAuthListDisablePostAuthorize * false</i> (If implemented) |
| Memory State: A known invalid idToken is configured in the Local Authorization List |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|---|------|
| Charging Station | CSMS |
| <p>1. Execute Reusable State <i>Authorized</i> for <Configured invalid IdToken fields> (idTokenInfo.status Accepted) <i>Note: The configured invalid idToken fields are being used, however the Test system responds with status Accepted to simulate an outdated local authorization list entry. In the mean time the idToken became valid.</i></p> | |
| <p>2. Execute Reusable State <i>EVConnectedPreSession</i></p> | |
| <p>3. Execute Reusable State <i>EnergyTransferStarted</i></p> | |

| |
|--|
| Tool validations |
| - N/a |
| Post scenario validations: - Energy is transferred |

TC_C_58_CS: Online authorization through local authorization list - LocalAuthListDisablePostAuthorize

| | |
|-------------------|--|
| Test case name | Online authorization through local authorization list - LocalAuthListDisablePostAuthorize |
| Test case Id | TC_C_58_CS |
| Use case Id(s) | C14 |
| Requirement(s) | C14_FR_03 |
| System under test | Charging Station |
| Description | This test case describes how to authorize an IdToken via the Local Authorization List while the Charging Station is online. While online the Charging Station can then locally authorize the IdToken, and is not required to send an AuthorizeRequest for a known IdToken. For an IdToken that does not have status "Accepted" the Charging Station will still send an AuthorizeRequest to check against the most recent status from CSMS. However, when the setting LocalAuthListDisablePostAuthorize is set to true, then the Charging Station will not do this. |
| Purpose | To verify that the Charging Station will not send an AuthorizeRequest for an idToken which has status "Invalid" in its local authorization list. |
| Prerequisite(s) | <ul style="list-style-type: none"> - LocalAuthListCtrlr.Available is implemented with value <i>true</i> AND - LocalAuthListCtrlr.DisablePostAuthorize is implemented AND - The Charging Station must support an authorization method other than NoAuthorization or Central |

| |
|--|
| Before (Preparations) |
| Configuration State: LocalAuthListCtrlr.Enabled is <i>true</i> (If implemented) AuthCtrlr.LocalPreAuthorize is <i>true</i> (If implemented) LocalAuthListCtrlr.DisablePostAuthorize <i>true</i> |
| Memory State: <i>IdTokenLocalAuthList</i> for <Configured invalid idtoken fields> |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|---|---|
| Charging Station | CSMS |
| <u>Manual Action</u> : Present Invalid idToken which is already stored in the Local Authorization List. | |
| <u>Note</u> : This step is executed if configured scenario is Local | |
| 2. The Charging Station responds with a RequestStartTransactionResponse | 1. The Test System sends a RequestStartTransactionRequest with idToken.idToken <Configured invalid_idtoken_idtoken> idToken.type <Configured invalid_idtoken_type> evseld <Configured evseld> <u>Note</u> : This step is executed if configured scenario is Remote |
| 3. The Charging Station does NOT send a AuthorizeRequest | |
| 4. Execute Reusable State <i>EVConnectedPreSession</i> | |
| 5. The Charging Station does NOT start charging. | |

| |
|--|
| Tool validations |
| * Step 3: Check that Charging Station does NOT send an AuthorizeRequest and authorization is refused. |
| Post scenario validations: The Charging Station does NOT start charging. |

TC_C_32_CS: Store Authorization Data in the Authorization Cache - Persistent over reboot

| | |
|--------------------------|---|
| Test case name | Store Authorization Data in the Authorization Cache - Persistent over reboot |
| Test case Id | TC_C_32_CS |
| Use case Id(s) | C10 |
| Requirement(s) | C10_FR_02 |
| System under test | Charging Station |
| Description | This test case covers how the Charging Station autonomously stores a record of previously presented identifiers that have been successfully authorized by the CSMS in the Authorization Cache. (Successfully meaning: a response received on a message containing an IdToken) |
| Purpose | To verify if the Charging Station is able to store the identifiers persistent over reboot according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | The Charging Station has an authorization cache AND Authorization cache is stored in the non-volatile memory AND the Charging Station supports at least one of the following local start authorization options: C-30, C-31, C-32, C-34 OR supports at least one of the following remote start authorization options: C-36, C-37 |

| |
|---|
| Before (Preparations) |
| Configuration State: AuthCacheEnabled is <i>true</i> (If implemented) LocalPreAuthorize is <i>true</i> |
| Memory State: <i>IdTokenCached</i> for <Configured valid IdToken fields> |
| Reusable State(s): N/a |

| | |
|--|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>Booted</i> | |
| 2. Execute Reusable State <i>Authorized</i> (Cached idToken) | |
| 3. Execute Reusable State <i>EVConnectedPreSession</i> | |
| 4. Execute Reusable State <i>EnergyTransferStarted</i> | |

| |
|--|
| Tool validations |
| N/a |
| Post scenario validations: - N/a |

NOTE | If the Charging Station supports ISO15118, this testcase needs to be executed using EIM.

TC_C_33_CS: Store Authorization Data in the Authorization Cache - Update on AuthorizeResponse

| | |
|--------------------------|---|
| Test case name | Store Authorization Data in the Authorization Cache - Update on AuthorizeResponse |
| Test case Id | TC_C_33_CS |
| Use case Id(s) | C10 |
| Requirement(s) | C10.FR.04, C12.FR.06 |
| System under test | Charging Station |
| Description | This test case covers how the Charging Station autonomously stores a record of previously presented identifiers that have been successfully authorized by the CSMS in the Authorization Cache. (Successfully meaning: a response received on a message containing an IdToken) |
| Purpose | To verify if the Charging Station is able to store the identifiers correctly upon an AuthorizeResponse according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | The Charging Station has an authorization cache AND the Charging Station supports at least one of the following local start authorization options: C-30, C-31, C-32, C-34 OR supports at least one of the following remote start authorization options: C-36, C-37 |

| |
|---|
| Before (Preparations) |
| Configuration State: AuthCacheEnabled is <i>true</i> (If implemented) LocalPreAuthorize is <i>true</i> LocalAuthListEnabled is <i>true</i> |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>IdTokenCached</i> for <Configured valid IdToken fields> | |
| 2. Execute Reusable State <i>Authorized</i> (Cached idToken) | |
| 3. Execute Reusable State <i>EVConnectedPreSession</i> | |
| 4. Execute Reusable State <i>EnergyTransferStarted</i> | |

| |
|--|
| Tool validations |
| N/a |
| Post scenario validations: - N/a |

NOTE | If the Charging Station supports ISO15118, this testcase needs to be executed using EIM.

TC_C_34_CS: Store Authorization Data in the Authorization Cache - Update on TransactionResponse

| | |
|-------------------|---|
| Test case name | Store Authorization Data in the Authorization Cache - Update on TransactionResponse |
| Test case Id | TC_C_34_CS |
| Use case Id(s) | C10 |
| Requirement(s) | C10.FR.05, C12.FR.06 |
| System under test | Charging Station |
| Description | This test case covers how the Charging Station autonomously stores a record of previously presented identifiers that have been successfully authorized by the CSMS in the Authorization Cache. (Successfully meaning: a response received on a message containing an IdToken) |
| Purpose | To verify if the Charging Station is able to store the identifiers correctly upon an TransactionResponse according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | The Charging Station has an authorization cache AND the Charging Station supports at least one of the following local start authorization options: C-30, C-31, C-32, C-34 OR supports at least one of the following remote start authorization options: C-36, C-37 |

| |
|---|
| Before (Preparations) |
| Configuration State: AuthCacheEnabled is <i>true</i> (If implemented) LocalPreAuthorize is <i>true</i> LocalAuthListEnabled is <i>true</i> StopTxOnInvalidId is <i>true</i> MaxEnergyOnInvalidId is 0 |
| Memory State: IdTokenCached for <Configured valid IdToken fields> |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|---|------|
| Charging Station | CSMS |
| 1. Execute Reusable State <i>Authorized</i> (Cached idToken, idTokenInfo.status invalid) | |
| 2. Execute Reusable State <i>Deauthorized</i> | |
| 3. Execute Reusable State <i>EVDisconnected</i> | |
| 4. Execute Reusable State <i>ParkingBayUnoccupied</i> | |
| 5. Execute Reusable State <i>ParkingBayOccupied</i> | |
| 6. Execute Reusable State <i>Authorized</i> (idTokenInfo.status invalid) | |

| |
|--|
| Tool validations |
| N/a |
| Post scenario validations: - N/a |

NOTE | If the Charging Station supports ISO15118, this testcase needs to be executed using EIM.

TC_C_36_CS: Store Authorization Data in the Authorization Cache - AuthCacheCtrlr.LocalPreAuthorize = false

| | |
|-------------------|---|
| Test case name | Store Authorization Data in the Authorization Cache - AuthCacheCtrlr.LocalPreAuthorize = false |
| Test case Id | TC_C_36_CS |
| Use case Id(s) | C10 |
| Requirement(s) | C10_FR_11 |
| System under test | Charging Station |
| Description | This test case covers how the Charging Station autonomously stores a record of previously presented identifiers that have been successfully authorized by the CSMS in the Authorization Cache. (Successfully meaning: a response received on a message containing an IdToken) |
| Purpose | To verify if the Charging Station is able to ignore the Authorization Cache feature when LocalPreAuthorize is set to false according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | The Charging Station has an authorization cache AND the Charging Station supports at least one of the following local start authorization options: C-30, C-31, C-32, C-34 OR supports at least one of the following remote start authorization options: C-36, C-37 |

| |
|---|
| Before (Preparations) |
| Configuration State: AuthCacheEnabled is true LocalPreAuthorize is false |
| Memory State: IdTokenCached for <Configured valid IdToken fields> |
| Reusable State(s): N/a |

| | |
|---|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>Authorized</i> (idTokenInfo.status invalid) | |

| |
|--|
| Tool validations |
| N/a |
| Post scenario validations: - N/a |

NOTE | If the Charging Station supports ISO15118, this testcase needs to be executed using EIM.

TC_C_37_CS: Clear Authorization Data in Authorization Cache - Accepted

| | |
|-------------------|---|
| Test case name | Clear Authorization Data in Authorization Cache - Accepted |
| Test case Id | TC_C_37_CS |
| Use case Id(s) | C11 |
| Requirement(s) | C11.FR.01, C11.FR.02, C11.FR.03 |
| System under test | Charging Station |
| Description | This test case covers how the Charging Station autonomously stores a record of previously presented identifiers that have been successfully authorized by the CSMS in the Authorization Cache. (Successfully meaning: a response received on a message containing an IdToken) |
| Purpose | To verify if the Charging Station is able to clear all identifiers from the Authorization Cache according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | The Charging Station has an authorization cache AND the Charging Station supports at least one of the following local start authorization options: C-30, C-31, C-32, C-34 OR supports at least one of the following remote start authorization options: C-36, C-37 |

| |
|--|
| Before (Preparations) |
| Configuration State: AuthCacheEnabled is <i>true</i> (If implemented) |
| Memory State: <i>IdTokenCached</i> for <Configured valid IdToken fields> |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a ClearCacheResponse | 1. The Test System sends a ClearCacheRequest |
| 3. Execute Reusable State <i>Authorized</i> | |
| 4. Execute Reusable State <i>EVConnectedPreSession</i> | |
| 5. Execute Reusable State <i>EnergyTransferStarted</i> | |

| |
|---|
| Tool validations |
| * Step 2: Message ClearCacheResponse - status <i>Accepted</i> |
| Post scenario validations: - N/a |

NOTE | If the Charging Station supports ISO15118, this testcase needs to be executed using EIM.

TC_C_38_CS: Clear Authorization Data in Authorization Cache - Rejected

| | |
|-------------------|---|
| Test case name | Clear Authorization Data in Authorization Cache - Rejected |
| Test case Id | TC_C_38_CS |
| Use case Id(s) | C11 |
| Requirement(s) | C11.FR.01, C11.FR.02, C11.FR.04 |
| System under test | Charging Station |
| Description | This test case covers how the Charging Station autonomously stores a record of previously presented identifiers that have been successfully authorized by the CSMS in the Authorization Cache. (Successfully meaning: a response received on a message containing an IdToken) |
| Purpose | To verify if the Charging Station is able to correctly respond on a request from the CSMS to clear all identifiers from the Authorization Cache while the feature is disabled according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | The Charging Station has an authorization cache AND the Charging Station supports at least one of the following local start authorization options: C-30, C-31, C-32, C-34 OR supports at least one of the following remote start authorization options: C-36, C-37 |

| |
|--|
| Before (Preparations) |
| Configuration State: AuthCacheEnabled is <i>false</i> (If implemented) |
| Memory State: <i>IdTokenCached</i> for <Configured valid IdToken fields> |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a ClearCacheResponse | 1. The Test System sends a ClearCacheRequest |

| |
|---|
| Tool validations |
| * Step 2: Message ClearCacheResponse - status <i>Rejected</i> |
| Post scenario validations: - N/a |

NOTE | If the Charging Station supports ISO15118, this testcase needs to be executed using EIM.

TC_C_39_CS: Authorization by GroupId - Success

| | |
|-------------------|---|
| Test case name | Authorization by GroupId - Success |
| Test case Id | TC_C_39_CS |
| Use case Id(s) | C09 |
| Requirement(s) | C09_FR_02, C09_FR_03, C09_FR_05 |
| System under test | Charging Station |
| Description | This test case covers how a Charging Station can authorize an action for an EV Driver based on GroupId information. This could for example be used if 2 people regularly use the same EV: they can use their own IdToken (e.g. RFID card), and can deauthorize transactions that were started with the other idToken (with the same GroupId). |
| Purpose | To verify if the Charging Station is able to correctly handle the Authorization of idTokens with the same GroupId according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | - The Charging Station supports authorization methods other than NoAuthorization |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State(s):

State is *EVConnectedPreSession*

Main (Test scenario)

| | |
|--|---|
| Charging Station | CSMS |
| <u>Manual Action:</u> Present valid idToken with <Configured GroupId> | |
| 1. The Charging Station sends an AuthorizeRequest | 2. The Test System responds with an AuthorizeResponse with - idTokenInfo.status <i>Accepted</i> - idTokenInfo.groupIdToken.idToken <Configured groupIdToken> |
| 3. The Charging Station sends a TransactionEventRequest <u>Note(s):</u> - This step needs to be executed when TxStartPoint contains <i>Authorized</i> OR the transaction already started. So in the case TxStartPoint contains <i>ParkingBayOccupancy</i> | 4. The Test System responds with a TransactionEventResponse with - idTokenInfo.status <i>Accepted</i> - idTokenInfo.groupIdToken.idToken <Configured groupIdToken> |
| 5. Execute Reusable State <i>EnergyTransferStarted</i> | |
| <u>Manual Action:</u> Present other valid idToken with <Configured GroupId> | |
| 6. The Charging Station sends an AuthorizeRequest | 7. The Test System responds with an AuthorizeResponse with - idTokenInfo.status <i>Accepted</i> - idTokenInfo.groupIdToken.idToken <Configured groupIdToken> |
| 8. The Charging Station sends a TransactionEventRequest | 9. The Test System responds with a TransactionEventResponse with - idTokenInfo.status <i>Accepted</i> - idTokenInfo.groupIdToken.idToken <Configured groupIdToken> |
| 10. Execute Reusable State <i>EVConnectedPostSession</i> | |
| 11. Execute Reusable State <i>EVDDisconnected</i> | |
| 12. Execute Reusable State <i>ParkingBayUnoccupied</i> | |

| Tool validations |
|---|
| <div><div>* Step 1:</div><div>Message AuthorizeRequest</div><div><div>- idToken.idToken <Configured valid_idtoken_idtoken></div><div>- idToken.type <Configured valid_idtoken_type></div></div><div>* Step 3:</div><div>Message TransactionEventRequest</div><div><div>- triggerReason <i>Authorized</i></div><div>- idToken.idToken <Configured valid_idtoken_idtoken></div><div>- idToken.type <Configured valid_idtoken_type></div></div><div>if transaction was already started</div><div><div>- eventType <i>Updated</i></div></div><div>else</div><div><div>- eventType <i>Started</i></div></div><div>* Step 6:</div><div>Message AuthorizeRequest</div><div><div>- idToken.idToken <Configured valid_idtoken_idtoken></div><div>- idToken.type <Configured valid_idtoken_type></div></div><div>* Step 8:</div><div>Message TransactionEventRequest</div><div><div>- triggerReason <i>StopAuthorized</i></div><div>- idToken.idToken <Configured valid_idtoken_idtoken></div><div>- idToken.type <Configured valid_idtoken_type></div></div></div> |
| <div><div>Post scenario validations:</div><div>- N/a</div></div> |

TC_C_40_CS: Authorization by GroupId - Success with Local Authorization List

| | |
|-------------------|---|
| Test case name | Authorization by GroupId - Success with Local Authorization List |
| Test case Id | TC_C_40_CS |
| Use case Id(s) | C09 |
| Requirement(s) | C09_FR_02, C09_FR_03, C09_FR_07 |
| System under test | Charging Station |
| Description | This test case covers how a Charging Station can authorize an action for an EV Driver based on GroupId information. This could for example be used if 2 people regularly use the same EV: they can use their own IdToken (e.g. RFID card), and can deauthorize transactions that were started with the other idToken (with the same GroupId). |
| Purpose | To verify if the Charging Station is able to correctly handle the Authorization of idTokens with the same GroupId when stored in the Local Authorization List according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | <ul style="list-style-type: none"> - LocalAuthListCtrlr.LocalAuthListAvailable is implemented with value true - The Charging Station supports authorization methods other than NoAuthorization |

Before (Preparations)

Configuration State:

LocalAuthListEnabled is *true* (If implemented)

LocalPreAuthorize is *true* (If implemented)

Memory State:

Two known valid idTokens are configured in the Local Authorization List with the same GroupId

Reusable State(s):

State is *EVConnectedPreSession*

Main (Test scenario)

| Charging Station | CSMS |
|--|---|
| <u>Manual Action:</u> Present valid idToken with <Configured GroupId> which is configured in the Local Authorization List | |
| 1. The Charging Station sends a TransactionEventRequest <u>Note(s):</u> - This step needs to be executed when TxStartPoint contains Authorized OR the transaction already started. So in the case TxStartPoint contains ParkingBayOccupancy | 2. The Test System responds with a TransactionEventResponse with - idTokenInfo.status Accepted - idTokenInfo.groupIdToken.idToken <Configured groupIdToken> |
| 3. Execute Reusable State <i>EnergyTransferStarted</i> | |
| <u>Manual Action:</u> Present valid idToken2 with <Configured GroupId> which is configured in the Authorization Cache | |
| 4. Execute Reusable State <i>StopAuthorized</i> | |
| 5. Execute Reusable State <i>EVConnectedPostSession</i> | |
| 6. Execute Reusable State <i>EVDisconnected</i> | |
| 7. Execute Reusable State <i>ParkingBayUnoccupied</i> | |

| Tool validations |
|--|
| <div>* Step 1: Message TransactionEventRequest - triggerReason <i>Authorized</i> - idToken.idToken <i><Configured valid_idtoken_idtoken></i> - idToken.type <i><Configured valid_idtoken_type></i> if transaction was already started - eventType <i>Updated</i> else - eventType <i>Started</i> * Step 4: Message TransactionEventRequest - triggerReason <i>StopAuthorized</i> - idToken.idToken <i><Configured valid_idtoken_idtoken></i> - idToken.type <i><Configured valid_idtoken_type></i> - eventType <i>Updated</i></div> |
| <div>Post scenario validations: - N/a</div> |

TC_C_41_CS: Authorization by GroupId - Success with Authorization Cache

| | |
|-------------------|---|
| Test case name | Authorization by GroupId - Success with Authorization Cache |
| Test case Id | TC_C_41_CS |
| Use case Id(s) | C09 |
| Requirement(s) | C09_FR_02, C09_FR_03, C09_FR_07 |
| System under test | Charging Station |
| Description | This test case covers how a Charging Station can authorize an action for an EV Driver based on GroupId information. This could for example be used if 2 people regularly use the same EV: they can use their own IdToken (e.g. RFID card), and can deauthorize transactions that were started with the other idToken (with the same GroupId). |
| Purpose | To verify if the Charging Station is able to correctly handle the Authorization of idTokens with the same GroupId when stored in the Authorization Cache according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | <ul style="list-style-type: none"> - AuthCacheCtrlr.Available is implemented with value true - The Charging Station supports authorization methods other than NoAuthorization |

Before (Preparations)

Configuration State:

AuthCacheEnabled is *true* (If implemented)

LocalPreAuthorize is *true* (If implemented)

Memory State:

IdTokenCached for <Configured valid IdToken fields>

IdTokenCached for <Configured valid IdToken2 fields>

Reusable State(s):

State is *EVConnectedPreSession*

Main (Test scenario)

| Charging Station | CSMS |
|--|---|
| <u>Manual Action:</u> Present valid idToken with <Configured GroupId> which is configured in the Authorization Cache | |
| 1. The Charging Station sends a TransactionEventRequest <u>Note(s):</u> - This step needs to be executed when TxStartPoint contains Authorized OR the transaction already started. So in the case TxStartPoint contains ParkingBayOccupancy | 2. The Test System responds with a TransactionEventResponse with - idTokenInfo.status Accepted - idTokenInfo.groupIdToken.idToken <Configured groupIdToken> |
| 3. Execute Reusable State <i>EnergyTransferStarted</i> | |
| <u>Manual Action:</u> Present valid idToken2 with <Configured GroupId> which is configured in the Authorization Cache | |
| 4. Execute Reusable State <i>StopAuthorized</i> | |
| 5. Execute Reusable State <i>EVConnectedPostSession</i> | |
| 6. Execute Reusable State <i>EVDisconnected</i> | |
| 7. Execute Reusable State <i>ParkingBayUnoccupied</i> | |

| Tool validations |
|--|
| <div>* Step 1: Message TransactionEventRequest - triggerReason <i>Authorized</i> - idToken.idToken <i><Configured valid_idtoken_idtoken></i> - idToken.type <i><Configured valid_idtoken_type></i> if transaction was already started - eventType <i>Updated</i> else - eventType <i>Started</i> * Step 4: Message TransactionEventRequest - triggerReason <i>StopAuthorized</i> - idToken.idToken <i><Configured valid_idtoken_idtoken></i> - idToken.type <i><Configured valid_idtoken_type></i></div> |
| <div>Post scenario validations: - N/a</div> |

NOTE

If the Charging Station supports ISO15118, this testcase needs to be executed using EIM.

TC_C_42_CS: Authorization by GroupId - Not stopped by GroupId

| | |
|-------------------|---|
| Test case name | Authorization by GroupId - Not stopped by GroupId |
| Test case Id | TC_C_42_CS |
| Use case Id(s) | C09 |
| Requirement(s) | C09_FR_11 |
| System under test | Charging Station |
| Description | This test case covers how a Charging Station can authorize an action for an EV Driver based on GroupId information. This could for example be used if 2 people regularly use the same EV: they can use their own IdToken (e.g. RFID card), and can deauthorize transactions that were started with the other idToken (with the same GroupId). |
| Purpose | To verify if the Charging Station is able to correctly handle the Authorization of idTokens with the same GroupId, while one of them is invalid, according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | - The Charging Station supports authorization methods other than NoAuthorization |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EVConnectedPreSession</i> |

| Main (Test scenario) | |
|--|---|
| Charging Station | CSMS |
| <u>Manual Action:</u> Present valid idToken with <Configured GroupId> | |
| 1. The Charging Station sends an AuthorizeRequest | 2. The Test System responds with an AuthorizeResponse with - idTokenInfo.status <i>Accepted</i> - idTokenInfo.groupIdToken.idToken <Configured groupIdToken> |
| 3. The Charging Station sends a TransactionEventRequest <u>Note(s):</u> - This step needs to be executed when TxStartPoint contains Authorized OR the transaction already started. So in the case TxStartPoint contains ParkingBayOccupancy | 4. The Test System responds with a TransactionEventResponse with - idTokenInfo.status <i>Accepted</i> - idTokenInfo.groupIdToken.idToken <Configured groupIdToken> |
| 5. Execute Reusable State <i>EnergyTransferStarted</i> | |
| <u>Manual Action:</u> Present invalid idToken with <Configured GroupId> | |
| 6. The Charging Station sends an AuthorizeRequest | 7. The Test System responds with an AuthorizeResponse with - idTokenInfo.status <i>Invalid</i> - idTokenInfo.groupIdToken.idToken <Configured groupIdToken> |
| <u>Note(s):</u> Test System will wait to see if CS indeed doesn't send a TransactionEventRequest | |

| Tool validations |
|---|
| <p>* Step 1:</p> <p>Message AuthorizeRequest</p> <ul style="list-style-type: none">- idToken.idToken <Configured valid_idtoken_idtoken>- idToken.type <Configured valid_idtoken_type> <p>* Step 3:</p> <p>Message TransactionEventRequest</p> <ul style="list-style-type: none">- triggerReason <i>Authorized</i>- idToken.idToken <Configured valid_idtoken_idtoken>- idToken.type <Configured valid_idtoken_type> <p>if transaction was already started</p> <ul style="list-style-type: none">- eventType <i>Updated</i> <p>else</p> <ul style="list-style-type: none">- eventType <i>Started</i> <p>* Step 6:</p> <p>Message AuthorizeRequest</p> <ul style="list-style-type: none">- idToken.idToken <Configured valid_idtoken_idtoken>- idToken.type <Configured valid_idtoken_type> |
| <p>Post scenario validations:</p> <ul style="list-style-type: none">- The energy transfer is not stopped |

TC_C_43_CS: Authorization by GroupId - Invalid status with Local Authorization List

| | |
|-------------------|---|
| Test case name | Authorization by GroupId - Invalid status with Local Authorization List |
| Test case Id | TC_C_43_CS |
| Use case Id(s) | C09 |
| Requirement(s) | C09_FR_02, C09_FR_03, C09_FR_07 |
| System under test | Charging Station |
| Description | This test case covers how a Charging Station can authorize an action for an EV Driver based on GroupId information. This could for example be used if 2 people regularly use the same EV: they can use their own IdToken (e.g. RFID card), and can deauthorize transactions that were started with the other idToken (with the same GroupId). |
| Purpose | To verify if the Charging Station is able to correctly handle the Authorization of idTokens with the same GroupId when stored in the Local Authorization List, but one of them is invalid, according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | <ul style="list-style-type: none"> - LocalAuthListCtrlr.LocalAuthListAvailable is implemented with value true - The Charging Station supports authorization methods other than NoAuthorization |

| |
|--|
| Before (Preparations) |
| Configuration State: LocalAuthListEnabled is <i>true</i> (If implemented) LocalPreAuthorize is <i>true</i> (If implemented) |
| Memory State: Two known idTokens are configured in the Local Authorization List with the same GroupId, one is valid and one is invalid. |
| Reusable State(s): State is <i>EVConnectedPreSession</i> |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action:</u> Present valid idToken with <Configured GroupId> which is configured in the Local Authorization List | |
| 1. The Charging Station sends a TransactionEventRequest <u>Note(s):</u> - This step needs to be executed when TxStartPoint contains Authorized OR the transaction already started. So in the case TxStartPoint contains ParkingBayOccupancy | 2. The Test System responds with a TransactionEventResponse with - idTokenInfo.status Accepted - idTokenInfo.groupIdToken.idToken <Configured groupIdToken> |
| 3. Execute Reusable State <i>EnergyTransferStarted</i> | |
| <u>Manual Action:</u> Present invalid idToken with <Configured GroupId> which is configured in the Local Authorization List | |
| 4. The Charging Station sends an AuthorizeRequest | 5. The Test System responds with an AuthorizeResponse with - idTokenInfo.status Invalid - idTokenInfo.groupIdToken.idToken <Configured groupIdToken> |
| <u>Note(s):</u> Test System will wait to see if CS indeed doesn't send a TransactionEventRequest | |

| Tool validations |
|---|
| <p>* Step 1:</p> <p>Message TransactionEventRequest</p> <ul style="list-style-type: none">- triggerReason <i>Authorized</i>- idToken.idToken <i><Configured valid_idtoken_idtoken></i>- idToken.type <i><Configured valid_idtoken_type></i> <p>if transaction was already started</p> <ul style="list-style-type: none">- eventType <i>Updated</i> <p>else</p> <ul style="list-style-type: none">- eventType <i>Started</i> <p>* Step 4:</p> <p>Message AuthorizeRequest</p> <ul style="list-style-type: none">- idToken.idToken <i><Configured valid_idtoken_idtoken></i>- idToken.type <i><Configured valid_idtoken_type></i> <p>* Step 6:</p> <p>Message TransactionEventRequest</p> <ul style="list-style-type: none">- triggerReason <i>StopAuthorized</i>- idToken.idToken <i><Configured valid_idtoken_idtoken></i>- idToken.type <i><Configured valid_idtoken_type></i>- eventType <i>Updated</i> |
| <p>Post scenario validations:</p> <ul style="list-style-type: none">- N/a |

TC_C_44_CS: Authorization by GroupId - Invalid status with Authorization Cache

| | |
|-------------------|---|
| Test case name | Authorization by GroupId - Invalid status with Authorization Cache |
| Test case Id | TC_C_44_CS |
| Use case Id(s) | C09 |
| Requirement(s) | C09_FR_02, C09_FR_03, C09_FR_07 |
| System under test | Charging Station |
| Description | This test case covers how a Charging Station can authorize an action for an EV Driver based on GroupId information. This could for example be used if 2 people regularly use the same EV: they can use their own IdToken (e.g. RFID card), and can deauthorize transactions that were started with the other idToken (with the same GroupId). |
| Purpose | To verify if the Charging Station is able to correctly handle the Authorization of idTokens with the same GroupId when stored in the Authorization Cache, but one of them is invalid, according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | <ul style="list-style-type: none"> - AuthCacheCtrlr.Available is implemented with value true - The Charging Station supports authorization methods other than NoAuthorization |

Before (Preparations)

Configuration State:

AuthCacheEnabled is *true* (If implemented)

LocalPreAuthorize is *true* (If implemented)

AuthCacheCtrlrDisablePostAuthorize is *false* (If implemented)

Memory State:

IdTokenCached for <Configured valid IdToken fields>

IdTokenCached for <Configured invalid IdToken fields>

Reusable State(s):

State is *EVConnectedPreSession*

Main (Test scenario)

| | |
|--|--|
| Charging Station | CSMS |
| <u>Manual Action:</u> Present valid idToken with <Configured GroupId> which is configured in the Authorization Cache | |
| 1. The Charging Station sends a TransactionEventRequest <u>Note(s):</u> - This step needs to be executed when TxStartPoint contains Authorized OR the transaction already started. So in the case TxStartPoint contains ParkingBayOccupancy | 2. The Test System responds with a TransactionEventResponse with - idTokenInfo.status Accepted - idTokenInfo.groupIdToken.idToken <Configured groupIdToken> |
| 3. Execute Reusable State <i>EnergyTransferStarted</i> <u>Manual Action:</u> Present invalid idToken with <Configured GroupId> which is configured in the Authorization Cache | |
| 4. The Charging Station sends an AuthorizeRequest | 5. The Test System responds with an AuthorizeResponse with - idTokenInfo.status Invalid - idTokenInfo.groupIdToken.idToken <Configured groupIdToken> |
| <u>Note(s):</u> Test System will wait to see if CS indeed doesn't send a TransactionEventRequest | |

| Tool validations |
|--|
| <div>* Step 1: Message TransactionEventRequest - triggerReason <i>Authorized</i> - idToken.idToken <i><Configured valid_idtoken_idtoken></i> - idToken.type <i><Configured valid_idtoken_type></i> if transaction was already started - eventType <i>Updated</i> else - eventType <i>Started</i> * Step 4: Message AuthorizeRequest - idToken.idToken <i><Configured invalid_idtoken_idtoken></i> - idToken.type <i><Configured invalid_idtoken_type></i></div> |
| <div>Post scenario validations: - N/a</div> |

NOTE

If the Charging Station supports ISO15118, this testcase needs to be executed using EIM.

TC_C_45_CS: Authorization by GroupId - Master pass - Not able to start transaction + groupId

| | |
|-------------------|---|
| Test case name | Authorization by GroupId - Master pass - Not able to start transaction + groupId |
| Test case Id | TC_C_45_CS |
| Use case Id(s) | C09 |
| Requirement(s) | C16.FR.03 |
| System under test | Charging Station |
| Description | This test case covers how a Charging Station can authorize an action for an EV Driver based on GroupId information. This could for example be used if 2 people regularly use the same EV: they can use their own IdToken (e.g. RFID card), and can deauthorize transactions that were started with the other idToken (with the same GroupId). |
| Purpose | To verify if the Charging Station is able to correctly handle the Authorization of an idToken with the same GroupId as the MasterPassGroupId according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The Charging station supports MasterPass feature. - The Charging Station supports authorization methods other than NoAuthorization |

| |
|--|
| Before (Preparations) |
| Configuration State: TxCtrlr.TxStartPoint should contain <i>Authorized</i> or <i>PowerPathClosed</i> and not contain <i>ParkingBayOccupancy</i> or <i>EVConnected</i> AuthCtrlr.MasterPassGroupId is <Configured MasterPassGroupId> |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action:</u> <i>Present configured masterpass idToken</i> | |
| 1. The Charging Station sends an AuthorizeRequest | 2. The Test System responds with an AuthorizeResponse with - idTokenInfo.status <i>Accepted</i> - idTokenInfo.groupIdToken.idToken <i><Configured masterPassGroupId></i> |
| <u>Note:</u> <i>The Charging Station will not authorize the transaction and send a TransactionEventRequest (in case of TxStartPoint Authorized).</i> | |
| 3. Execute Reusable State <i>EVConnectedPreSession</i> | |
| 4. The Charging Station will NOT send a TransactionEventRequest with chargingState <i>Charging</i> and triggerReason <i>ChargingStateChanged</i> | |

| |
|--|
| Tool validations |
| * Step 1: Message AuthorizeRequest - idToken.idToken <Configured valid_idtoken_idtoken> - idToken.type <Configured valid_idtoken_type> |
| Post scenario validations: - N/a |

TC_C_46_CS: Store Authorization Data in the Authorization Cache - AuthCacheLifeTime

| | |
|-------------------|---|
| Test case name | Store Authorization Data in the Authorization Cache - AuthCacheLifeTime |
| Test case Id | TC_C_46_CS |
| Use case Id(s) | C10 |
| Requirement(s) | C10_FR_08 |
| System under test | Charging Station |
| Description | This test case covers how the Charging Station autonomously stores a record of previously presented identifiers that have been successfully authorized by the CSMS in the Authorization Cache. (Successfully meaning: a response received on a message containing an IdToken) |
| Purpose | To verify if the Charging Station is able to correctly remove an idToken when this one is not reused again within the specified amount of time (AuthCacheLifeTime) according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | - AuthCacheCtrlr.Available is implemented with value true - Configuration variable AuthCacheLifeTime is implemented |

| |
|---|
| Before (Preparations) |
| Configuration State: AuthCacheLifeTime is <Configured TransactionDuration> AuthCacheCtrlr.LocalPreAuthorize is true (If implemented) |
| Memory State: IdTokenCached <Configured valid idtoken fields> |
| Reusable State(s): N/a |

| | |
|---|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Wait for <Configured Transaction Duration> seconds | |
| 2. Execute Reusable State Authorized | |

| |
|--|
| Tool validations |
| N/a |
| Post scenario validations: - N/a |

TC_C_47_CS: Stop Transaction with a Master Pass - With UI - All transactions

| | |
|-------------------|--|
| Test case name | Stop Transaction with a Master Pass - With UI - All transactions |
| Test case Id | TC_C_47_CS |
| Use case Id(s) | C16 |
| Requirement(s) | C16_FR_01 |
| System under test | Charging Station |
| Description | This test case covers how somebody with a Master Pass (User) can stop (selected) ongoing transactions, so the cable becomes unlocked. This Master Pass can be configured in: MasterPassGroupId. This could for example be usefull for Law Enforcement officials. |
| Purpose | To verify if the Charging Station is able to correctly stop all transactions when an idToken which has the MasterPass as GroupId is used and the user has selected to stop all transactions in the User Interface according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The Charging Station supports at least one authorization method described at the following Use cases; C01, C04. - Charging station has a User Interface - The Charging Station supports authorization methods other than NoAuthorization |

| |
|---|
| Before (Preparations) |
| Configuration State: AuthCtrlr.MastersPassGroupId is configured |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> for all EVSE |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action:</u> Present configured masterpass idToken | |
| 1. The Charging Station sends an AuthorizeRequest | 2. The Test System responds with an AuthorizeResponse with idTokenInfo.status <i>Accepted</i> idTokenInfo.groupIdToken.idToken <i><Configured masterPassGroupId></i> |
| <u>Manual Action:</u> Select to stop all transactions | |
| 3. The Charging Station sends a TransactionEventRequest for all EVSE | 4. The Test System responds with a TransactionEventResponse with idTokenInfo.status <i>Accepted</i> idTokenInfo.groupIdToken.idToken <i><Configured masterPassGroupId></i> for all EVSE |
| <u>Note(s):</u> The charging Station will repeat for all running transactions and can stop a transaction in more than one TransactionEventRequest. | |
| 5. Execute Reusable State <i>EVConnectedPostSession</i> for all EVSE | |
| 6. Execute Reusable State <i>EVDisconnected</i> for all EVSE | |
| 7. Execute Reusable State <i>ParkingBayUnoccupied</i> for all EVSE | |

| Tool validations |
|---|
| <p>* Step 1:</p> <p>Message AuthorizeRequest</p> <ul style="list-style-type: none">- idToken.idToken <Configured masterpass_idtoken_idtoken>- idToken.type <Configured masterpass_idtoken_type> <p>* Step 3:</p> <p>Message TransactionEventRequest</p> <ul style="list-style-type: none">- transactionInfo.stoppedReason <i>MasterPass</i> (in last TransactionEventRequest)- idToken omit or- idToken.idToken <Configured masterpass_idtoken_idtoken> and- idToken.type <Configured masterpass_idtoken_type> (once per stopped transaction)- eventType <i>Ended</i> (in last TransactionEventRequest) |
| <p>Post scenario validations:</p> <ul style="list-style-type: none">- N/a |

TC_C_48_CS: Stop Transaction with a Master Pass - With UI - With UI - Specific transactions

| | |
|-------------------|--|
| Test case name | Stop Transaction with a Master Pass - With UI - With UI - Specific transactions |
| Test case Id | TC_C_48_CS |
| Use case Id(s) | C16 |
| Requirement(s) | C16_FR_01 |
| System under test | Charging Station |
| Description | This test case covers how somebody with a Master Pass (User) can stop (selected) ongoing transactions, so the cable becomes unlocked. This Master Pass can be configured in: MasterPassGroupId. This could for example be usefull for Law Enforcement officials. |
| Purpose | To verify if the Charging Station is able to correctly stop a transaction when an idToken which has the MasterPass as GroupId is used and the user has selected to stop one transaction in the User Interface according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The Charging Station supports at least one authorization method described at the following Use cases; C01, C04. - Charging station has a User Interface - The Charging Station supports authorization methods other than NoAuthorization |

| |
|---|
| Before (Preparations) |
| Configuration State: AuthCtrlr.MastersPassGroupId is configured |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> for all EVSE |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action:</u> Present configured masterpass idToken | |
| 1. The Charging Station sends an AuthorizeRequest | 2. The Test System responds with an AuthorizeResponse with idTokenInfo.status Accepted idTokenInfo.groupIdToken.idToken <Configured masterPassGroupId> |
| <u>Manual Action:</u> Select to stop the transaction on EVSE 1 | |
| 3. The Charging Station sends a TransactionEventRequest | 4. The Test System responds with a TransactionEventResponse with idTokenInfo.status Accepted idTokenInfo.groupIdToken.idToken <Configured masterPassGroupId> |
| 5. Execute Reusable State EVConnectedPostSession | |
| 6. Execute Reusable State EVDisconnected | |
| 7. Execute Reusable State ParkingBayUnoccupied | |

| Tool validations |
|---|
| <p>* Step 1:</p> <p>Message AuthorizeRequest</p> <ul style="list-style-type: none">- idToken.idToken <Configured masterpass_idtoken_idtoken>- idToken.type <Configured masterpass_idtoken_type> <p>* Step 3:</p> <p>Message TransactionEventRequest</p> <ul style="list-style-type: none">- transactionInfo.stoppedReason <i>MasterPass</i>- idToken <i>omit</i> or- idToken.idToken <Configured masterpass_idtoken_idtoken> and- idToken.type <Configured masterpass_idtoken_type> |
| <p>Post scenario validations:</p> <ul style="list-style-type: none">- All other EVSE still transfer energy |

TC_C_49_CS: Stop Transaction with a Master Pass - Without UI

| | |
|-------------------|--|
| Test case name | Stop Transaction with a Master Pass - Without UI |
| Test case Id | TC_C_49_CS |
| Use case Id(s) | C16 |
| Requirement(s) | C16_FR_02 |
| System under test | Charging Station |
| Description | This test case covers how somebody with a Master Pass (User) can stop (selected) ongoing transactions, so the cable becomes unlocked. This Master Pass can be configured in: MasterPassGroupId. This could for example be usefull for Law Enforcement officials. |
| Purpose | To verify if the Charging Station is able to correctly stop all transactions when an idToken which has the MasterPass as GroupId is used and the Charging station does not have an User Interface according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The Charging Station supports at least one authorization method described at the following Use cases; C01, C04. - The Charging Station supports authorization methods other than NoAuthorization |

Before (Preparations)

Configuration State:
AuthCtrlr.MastersPassGroupId is configured

Memory State:
N/a

Reusable State(s):
State is *EnergyTransferStarted* for EVSEId 1 and EVSEId 2 if the Charging Station has more than one EVSE. With:
 - <Configured valid_idtoken> for EVSE 1
 - <Configured valid_idtoken2> for EVSE 2

Main (Test scenario)

| | |
|---|--|
| Charging Station | CSMS |
| <u>Manual Action:</u> Present configured masterpass idToken | |
| 1. The Charging Station sends an AuthorizeRequest | 2. The Test System responds with an AuthorizeResponse with idTokenInfo.status <i>Accepted</i> idTokenInfo.groupIdToken.idToken <Configured masterPassGroupId> |
| 3. The Charging Station sends a TransactionEventRequest for EVSE 1 (and 2) | 4. The Test System responds with a TransactionEventResponse with idTokenInfo.status <i>Accepted</i> idTokenInfo.groupIdToken.idToken <Configured masterPassGroupId> for EVSE 1 (and 2) |
| 5. Execute Reusable State <i>EVConnectedPostSession</i> for EVSE 1 (and 2) | |
| 6. Execute Reusable State <i>EVDisconnected</i> for EVSE 1 (and 2) | |
| 7. Execute Reusable State <i>ParkingBayUnoccupied</i> for EVSE 1 (and 2) | |

| Tool validations |
|--|
| <div>* Step 1: Message AuthorizeRequest - idToken.idToken <Configured masterpass_idtoken_idtoken> - idToken.type <Configured masterpass_idtoken_type> * Step 3: Message TransactionEventRequest - transactionInfo.stoppedReason <i>MasterPass</i> - idToken <i>omit</i> or - idToken.idToken <Configured masterpass_idtoken_idtoken> and - idToken.type <Configured masterpass_idtoken_type></div> |
| <div>Post scenario validations: - N/a</div> |

TC_C_21_CS: Offline authorization through local authorization list - Accepted

| | |
|-------------------|---|
| Test case name | Offline authorization through local authorization list - Accepted |
| Test case Id | TC_C_21_CS |
| Use case Id(s) | C13 |
| Requirement(s) | C13.FR.02 |
| System under test | Charging Station |
| Description | This test case describes how to authorize an IdToken via the Local Authorization List while the Charging Station is online. When offline the Charging Station can then locally authorize the IdToken. |
| Purpose | To verify if the Charging Station is able to authorize an idToken which has status "Accepted" in its local authorization list according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | - LocalAuthListCtrlr.LocalAuthListAvailable is implemented with value true - The Charging Station supports authorization methods other than NoAuthorization |

| |
|---|
| Before (Preparations) |
| Configuration State: LocalAuthListEnabled is <i>true</i> (If implemented) LocalPreAuthorize is <i>true</i> (If implemented) OfflineTxForUnknownIdEnabled is <i>false</i> (If implemented) LocalAuthorizeOffline is <i>true</i> |
| Memory State: <i>IdTokenLocalAuthList</i> for <Configured valid idtoken fields> |
| Reusable State(s): State is <i>StartOfflineTransaction</i> |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action</u> : Present idToken. | |
| <u>Manual Action</u> : Unplug cable. | |
| <u>Manual Action</u> : Drive out of parkingbay. | |
| 1. The Charging Stations sends a TransactionEventRequest <u>Note(s)</u> : - The Charging Station will empty its Transaction message queue. This will contain one or more TransactionEventRequest messages | 2. The Test System responds with a TransactionEventResponse <u>Note(s)</u> : - The Test System will respond to the TransactionEventRequest containing the idToken, with idtokenInfo.status Accepted |
| 3. Execute Reusable State <i>ParkingBayUnoccupied</i> | |

| |
|--|
| Tool validations |
| * Step 1: Message(s) before the StopAuthorize: TransactionEventRequests - offline must be <i>true</i> One of the Message(s): TransactionEventRequest - TriggerReason must be <i>StopAuthorized</i> |
| Post scenario validations: N/a |

TC_C_22_CS: Offline authorization through local authorization list - Invalid

| | |
|-------------------|--|
| Test case name | Offline authorization through local authorization list - Invalid |
| Test case Id | TC_C_22_CS |
| Use case Id(s) | C13 |
| Requirement(s) | C13.FR.02 |
| System under test | Charging Station |
| Description | This test case describes how to authorize an IdToken via the Local Authorization List while the Charging Station is online. When offline the Charging Station can then locally authorize the IdToken. |
| Purpose | To verify if the Charging Station is able to respond correctly to an idToken which has status "Invalid" in its local authorization list according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | <ul style="list-style-type: none"> - LocalAuthListCtrlr.LocalAuthListAvailable is implemented with value true - The Charging Station supports authorization methods other than NoAuthorization |

| |
|---|
| Before (Preparations) |
| Configuration State: LocalAuthListEnabled is <i>true</i> (If implemented) LocalPreAuthorize is <i>true</i> (If implemented) OfflineTxForUnknownIdEnabled is <i>false</i> (If implemented) LocalAuthorizeOffline is <i>true</i> OfflineThreshold is <i><Configured RetryBackOffWaitMinimum_duration> + 60.0</i> RetryBackOffWaitMinimum is <i><Configured RetryBackOffWaitMinimum_duration></i> RetryBackOffRandomRange is <i>0</i> |
| Memory State: <i>IdTokenLocalAuthList</i> for <i><Configured invalid idtoken fields></i> |
| Reusable State(s): N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Test System closes the WebSocket connection AND does not accept a reconnect. | |
| 2. <u>Manual Action</u> : Drive EV into parking bay. | |
| 3. <u>Manual Action</u> : Present idToken. | |
| 4. The Test System accepts reconnection attempt from the Charging Station. | |
| 5. The Charging Stations sends a TransactionEventRequest | 6. The Test System responds with a TransactionEventResponse |
| <u>Note(s)</u> : - The Charging Station will empty its Transaction message queue, this will only contain TransactionEventRequests if TxStartPoint was ParkingBayOccupancy and a Transaction was started. | |

| |
|---|
| Tool validations |
| * Step 5: Message TransactionEventRequest - offline must be <i>true</i> - TriggerReason must be <i>EVDetected</i> |
| Post scenario validations: N/a |

TC_C_23_CS: Offline authorization through local authorization list - Blocked

| | |
|-------------------|--|
| Test case name | Offline authorization through local authorization list - Blocked |
| Test case Id | TC_C_23_CS |
| Use case Id(s) | C13 |
| Requirement(s) | C13.FR.02 |
| System under test | Charging Station |
| Description | This test case describes how to authorize an IdToken via the Local Authorization List while the Charging Station is online. When offline the Charging Station can then locally authorize the IdToken. |
| Purpose | To verify if the Charging Station is able to respond correctly to an idToken which has status "Blocked" in its local authorization list according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | <ul style="list-style-type: none"> - LocalAuthListCtrlr.LocalAuthListAvailable is implemented with value true - The Charging Station supports authorization methods other than NoAuthorization |

| |
|---|
| Before (Preparations) |
| Configuration State: LocalAuthListEnabled is <i>true</i> (If implemented) LocalPreAuthorize is <i>true</i> (If implemented) OfflineTxForUnknownIdEnabled is <i>false</i> (If implemented) LocalAuthorizeOffline is <i>true</i> OfflineThreshold is <i><Configured RetryBackOffWaitMinimum_duration> + 60.0</i> RetryBackOffWaitMinimum is <i><Configured RetryBackOffWaitMinimum_duration></i> RetryBackOffRandomRange is <i>0</i> |
| Memory State: <i>IdTokenLocalAuthList</i> for <i><Configured blocked idtoken fields></i> |
| Reusable State(s): N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Test System closes the WebSocket connection AND does not accept a reconnect. | |
| 2. <u>Manual Action</u> : Drive EV into parking bay. | |
| 3. <u>Manual Action</u> : Present idToken. | |
| 4. The Test System accepts reconnection attempt from the Charging Station. | |
| 5. The Charging Stations sends a TransactionEventRequest | 6. The Test System responds with a TransactionEventResponse |
| <u>Note(s)</u> : - The Charging Station will empty its Transaction message queue, this will only contain TransactionEventRequests if TxStartPoint was ParkingBayOccupancy and a Transaction was started. | |

| |
|---|
| Tool validations |
| * Step 5: Message TransactionEventRequest - offline must be <i>true</i> - TriggerReason must be <i>EVDetected</i> |
| Post scenario validations: N/a |

TC_C_24_CS: Offline authorization through local authorization list - Expired

| | |
|-------------------|--|
| Test case name | Offline authorization through local authorization list - Expired |
| Test case Id | TC_C_24_CS |
| Use case Id(s) | C13 |
| Requirement(s) | C13.FR.02 |
| System under test | Charging Station |
| Description | This test case describes how to authorize an IdToken via the Local Authorization List while the Charging Station is online. When offline the Charging Station can then locally authorize the IdToken. |
| Purpose | To verify if the Charging Station is able to respond correctly to an idToken which has status "Expired" in its local authorization list according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | <ul style="list-style-type: none"> - LocalAuthListCtrlr.LocalAuthListAvailable is implemented with value true - The Charging Station supports authorization methods other than NoAuthorization |

| |
|--|
| Before (Preparations) |
| <p>Configuration State: LocalAuthListEnabled is <i>true</i> (If implemented) LocalPreAuthorize is <i>true</i> (If implemented) OfflineTxForUnknownIdEnabled is <i>false</i> (If implemented) LocalAuthorizeOffline is <i>true</i> OfflineThreshold is <i><Configured RetryBackOffWaitMinimum_duration> + 60.0</i> RetryBackOffWaitMinimum is <i><Configured RetryBackOffWaitMinimum_duration></i> RetryBackOffRandomRange is <i>0</i></p> <p>Memory State: <i>IdTokenLocalAuthList</i> for <i><Configured expired idtoken fields></i></p> <p>Reusable State(s): N/a</p> |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Test System closes the WebSocket connection AND does not accept a reconnect. | |
| 2. <u>Manual Action</u> : Drive EV into parking bay. | |
| 3. <u>Manual Action</u> : Present idToken. | |
| 4. The Test System accepts reconnection attempt from the Charging Station. | |
| 5. The Charging Stations sends a TransactionEventRequest | 6. The Test System responds with a TransactionEventResponse |
| <u>Note(s)</u> : - The Charging Station will empty its Transaction message queue, this will only contain TransactionEventRequests if TxStartPoint was ParkingBayOccupancy and a Transaction was started. | |

| |
|---|
| Tool validations |
| <p>* Step 5:</p> <p>Message TransactionEventRequest</p> <ul style="list-style-type: none"> - offline must be <i>true</i> - TriggerReason must be <i>EVDetected</i> <p>Post scenario validations: N/a</p> |

TC_C_25_CS: Offline authorization through local authorization list - Local Authorization List > Authorization Cache

| | |
|-------------------|---|
| Test case name | Offline authorization through local authorization list - Local Authorization List > Authorization Cache |
| Test case Id | TC_C_25_CS |
| Use case Id(s) | C13, C14 |
| Requirement(s) | C13.FR.01, C14.FR.01 |
| System under test | Charging Station |
| Description | This test case describes how to authorize an IdToken via the Local Authorization List while the Charging Station is online. When offline the Charging Station can then locally authorize the IdToken. |
| Purpose | To verify if the Charging Station does not start a transaction while being offline for an idToken that is stored in the cache, but also in the local authorization list as with status invalid. |
| Prerequisite(s) | <ul style="list-style-type: none"> - LocalAuthListCtrlr.LocalAuthListAvailable is implemented with value true - AuthCacheCtrlr.Available is implemented with value true - OfflineTxForUnknownIdEnabled is implemented. - The Charging Station supports authorization methods other than NoAuthorization |

Before (Preparations)

Configuration State:

LocalAuthListEnabled is *true* (If implemented)

LocalPreAuthorize is *true* (If implemented)

OfflineTxForUnknownIdEnabled is *true*

LocalAuthorizeOffline is *true*

StopTxOnInvalidId is *false*

OfflineThreshold is *<Configured RetryBackOffWaitMinimum_duration> + 60.0*

RetryBackOffWaitMinimum is *<Configured RetryBackOffWaitMinimum_duration>*

RetryBackOffRandomRange is *0*

Note:

<Configured RetryBackOffWaitMinimum_duration should be long enough to execute manual tasks>

Memory State:

IdTokenCached <Configured valid idtoken fields>

IdTokenLocalAuthList for <Configured valid idtoken fields, but set as invalid>

Reusable State(s):

State is *EVConnectedPreSession*

Main (Test scenario)

| | |
|---|------|
| Charging Station | CSMS |
| 1. The Test System closes the WebSocket connection AND does not accept a reconnect. | |
| <u>Manual Action</u> : Present idToken. | |
| <u>Note(s)</u> : The tool will wait for <i><Configured Transaction Duration></i> seconds. | |
| 2. The Test System accepts reconnection attempt from the Charging Station. | |
| 3. The Charging Station does NOT start a transaction. MeterValues are allowed. | |

Tool validations

| |
|-----------------------------------|
| N/a |
| Post scenario validations: N/a |

TC_C_26_CS: Offline Authorization - Unknown Id

| | |
|-------------------|--|
| Test case name | Offline Authorization - Unknown Id |
| Test case Id | TC_C_26_CS |
| Use case Id(s) | C15, C13 |
| Requirement(s) | C15.FR.02,C15.FR.06,C15.FR.08,C13.FR.04 |
| System under test | Charging Station |
| Description | The Charging Station is allowed to allow starting a transaction for unknown idTokens when offline and configured to do so. |
| Purpose | To verify if the Charging Station is able to start a transaction while being offline for an unknown idToken, when it is configured to do so. |
| Prerequisite(s) | <ul style="list-style-type: none"> - OfflineTxForUnknownIdEnabled is implemented. - The Charging Station supports authorization methods other than NoAuthorization |

| |
|---|
| Before (Preparations) |
| Configuration State: LocalAuthListEnabled is <i>true</i> (If implemented) LocalPreAuthorize is <i>true</i> (If implemented) OfflineTxForUnknownIdEnabled is <i>true</i> LocalAuthorizeOffline is <i>true</i> MaxEnergyOnInvalidId is <i>0</i> (If implemented) StopTxOnInvalidId is <i>false</i> |
| Memory State: N/a |
| Reusable State(s): State is <i>StartOfflineTransaction</i> |

| Main (Test scenario) | |
|---|---|
| Charging Station | CSMS |
| 1. The Charging Station notifies the CSMS about the current state of all connectors. | 2. The Test System responds accordingly. |
| 3. The Charging Stations sends a TransactionEventRequest <u>Note(s):</u> - The Charging Station will empty its Transaction message queue. This will contain one or more TransactionEventRequest messages | 4. The Test System responds with a TransactionEventResponse <u>Note(s):</u> - The Test System will respond to the TransactionEventRequest containing the idToken, with idtokenInfo.status Invalid |
| 5. Execute Reusable State <i>StopAuthorized</i> | |
| 6. Execute Reusable State <i>EVConnectedPostSession</i> | |
| 7. Execute Reusable State <i>EVDDisconnected</i> | |

| Tool validations |
|--|
| <p>* Step 1:</p> <p>Message: StatusNotificationRequest</p> <ul style="list-style-type: none">- connectorStatus must be <i>Occupied</i> <p>Message: NotifyEventRequest</p> <ul style="list-style-type: none">- eventData[0].trigger must be <i>Delta</i>- eventData[0].actualValue must be <i>Occupied</i>- eventData[0].component.name must be <i>Connector</i>- eventData[0].variable.name must be <i>AvailabilityState</i> <p>* Step 3:</p> <p>All Message(s): TransactionEventRequest</p> <ul style="list-style-type: none">- offline must be <i>true</i> <p>* Step 4:</p> <p>One of the Message(s): TransactionEventRequest</p> <ul style="list-style-type: none">- chargingState must be <i>SuspendedEVSE</i> |
| <p>Post scenario validations:</p> <p>N/a</p> |

TC_C_50_CS: Authorization using Contract Certificates 15118 - Online - Local contract certificate validation - Accepted

| | |
|-------------------|---|
| Test case name | Authorization using Contract Certificates 15118 - Online - Local contract certificate validation - Accepted |
| Test case Id | TC_C_50_CS |
| Use case Id(s) | C07 |
| Requirement(s) | C07.FR.01,C07.FR.02 |
| System under test | Charging Station |
| Description | The Charging Station is able to authorize with contract certificates when it supports ISO 15118. |
| Purpose | To verify if the Charging Station is able to authorize while locally validating the contract certificate. |
| Prerequisite(s) | N/a |

| |
|--|
| Before (Preparations) Configuration State: TxCtrlr.TxStartPoint contains one or more of <i>PowerPathClosed, Authorized, EVConnected, ParkingBayOccupancy</i> AuthCtrlr.AuthEnabled is <i>true</i> (If implemented AND ReadWrite) AuthCtrlr.DisableRemoteAuthorization is <i>false</i> (If implemented) For the ISO15118Ctrlr of the EVSE used in the PnC transaction: ISO15118Ctrlr.CentralContractValidationAllowed is <i>false</i> ISO15118Ctrlr.ContractCertificateInstallationEnabled is <i>true</i> ISO15118Ctrlr.V2GCertificateInstallationEnabled is <i>true</i> ISO15118Ctrlr.PnCEnabled is <i>true</i> Memory State: <i>CertificateInstalled</i> for certificateType <i>V2GRootCertificate</i> <i>CertificateInstalled</i> for certificateType <i>MORootCertificate</i> <i>RenewV2GChargingStationCertificate</i> Reusable State(s): State is <i>EVConnectedPreSession</i> |
|--|

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Charging Station sends an <i>AuthorizeRequest</i> <u>Note(s):</u> <i>-The test case should be robust enough to also handle a <i>GetCertificateStatusRequest</i></i> | 2. The Test System responds with a <i>AuthorizeResponse</i> with <i>idTokenInfo.status Accepted</i> and <i>certificateStatus = Accepted</i> |
| 3. The Charging Station sends a <i>TransactionEventRequest</i> | 4. The Test System responds with a <i>TransactionEventResponse</i> With <i>idTokenInfo.status Accepted</i> |
| 5. Execute Reusable State <i>EnergyTransferStarted</i> | |

| |
|---|
| Tool validations |
| * Step 1: Message: AuthorizeRequest - idToken.type must be <i>eMAID</i> - iso15118CertificateHashData is provided * Step 3: Message: TransactionEventRequest - triggerReason must be <i>Authorized</i> |
| Post scenario validations: N/a |

TC_C_51_CS: Authorization using Contract Certificates 15118 - Online - Local contract certificate validation - Rejected

| | |
|-------------------|---|
| Test case name | Authorization using Contract Certificates 15118 - Online - Local contract certificate validation - Rejected |
| Test case Id | TC_C_51_CS |
| Use case Id(s) | C07 |
| Requirement(s) | C07.FR.01,C07.FR.02 |
| System under test | Charging Station |
| Description | The Charging Station is able to authorize with contract certificates when it supports ISO 15118. |
| Purpose | To verify if the Charging Station is able to handle a rejected on an AuthorizeRequest, when authorizing using a contract certificate with an invalid EMAID. |
| Prerequisite(s) | N/a |

Before (Preparations)

Configuration State:

TxCtrlr.TxStartPoint contains one or more of *PowerPathClosed*, *Authorized*, *EVConnected*, *ParkingBayOccupancy*

AuthCtrlr.AuthEnabled is *true* (If implemented AND ReadWrite)

AuthCtrlr.DisableRemoteAuthorization is *false* (If implemented)

For the ISO15118Ctrlr of the EVSE used in the PnC transaction:

ISO15118Ctrlr.CentralContractValidationAllowed is *false*

ISO15118Ctrlr.PnCEnabled is *true*

Memory State:

CertificateInstalled for certificateType *V2GRootCertificate*

CertificateInstalled for certificateType *MORootCertificate*

RenewV2GChargingStationCertificate

Reusable State(s):

State is *EVConnectedPreSession*

Main (Test scenario)

| Charging Station | CSMS |
|---|--|
| 1. The Charging Station sends an AuthorizeRequest <u>Note(s):</u> <i>-The test case should be robust enough to also handle a GetCertificateStatusRequest</i> | 2. The Test System responds with a AuthorizeResponse with idTokenInfo.status <i>Invalid</i> and certificateStatus = <i>ContractCancelled</i> |

Tool validations

* Step 1:

Message: **AuthorizeRequest**

- **idToken.type** must be *eMAID*

- **iso15118CertificateHashData** is provided

Post scenario validations:

EV is not authorized and shall not charge:

Charging Station does not send **TransactionEventRequest** with:

- **triggerReason** = *Authorized* or **chargingState** = *Charging*

TC_C_52_CS: Authorization using Contract Certificates 15118 - Online - Central contract certificate validation - Accepted

| | |
|-------------------|---|
| Test case name | Authorization using Contract Certificates 15118 - Online - Central contract certificate validation - Accepted |
| Test case Id | TC_C_52_CS |
| Use case Id(s) | C07 |
| Requirement(s) | C07.FR.01,C07.FR.02,C07.FR.06 |
| System under test | Charging Station |
| Description | The Charging Station is able to authorize with contract certificates when it supports ISO 15118. |
| Purpose | To verify if the Charging Station is able to authorize, while not being able to locally validate the contract certificate and then send it to the CSMS. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The MO Root certificate that is needed to validate the EV Contract certificate must NOT be installed at the Charging Station. - The Charging Station supports central contract validation. |

| |
|---|
| Before (Preparations) |
| <p>Configuration State:</p> <p>TxCtrlr.TxStartPoint contains one or more of <i>PowerPathClosed</i>, <i>Authorized</i>, <i>EVConnected</i>, <i>ParkingBayOccupancy</i></p> <p>AuthCtrlr.AuthEnabled is <i>true</i> (If implemented AND ReadWrite)</p> <p>AuthCtrlr.DisableRemoteAuthorization is <i>false</i> (If implemented)</p> <p>For the ISO15118Ctrlr of the EVSE used for the PnC transaction:</p> <p>ISO15118Ctrlr.CentralContractValidationAllowed is <i>true</i></p> <p>ISO15118Ctrlr.ContractCertificateInstallationEnabled is <i>true</i></p> <p>ISO15118Ctrlr.V2GCertificateInstallationEnabled is <i>true</i></p> <p>ISO15118Ctrlr.PnCEnabled is <i>true</i></p> <p>Memory State:</p> <p><i>CertificateInstalled</i> for certificateType <i>V2GRootCertificate</i></p> <p><i>RenewV2GChargingStationCertificate</i></p> <p>Reusable State(s):</p> <p>State is <i>EVConnectedPreSession</i></p> |

| Main (Test scenario) | |
|--|---|
| Charging Station | CSMS |
| 1. The Charging Station sends an AuthorizeRequest <u>Note(s):</u> <i>-The test case should be robust enough to also handle a GetCertificateStatusRequest</i> | 2. The Test System responds with a AuthorizeResponse with idTokenInfo.status <i>Accepted</i> and certificateStatus = <i>Accepted</i> |
| 3. The Charging Station sends a TransactionEventRequest | 4. The Test System responds with a TransactionEventResponse With idTokenInfo.status <i>Accepted</i> |
| 5. Execute Reusable State <i>EnergyTransferStarted</i> | |

| |
|--|
| Tool validations |
| <p>* Step 1:</p> <p>Message: AuthorizeRequest</p> <ul style="list-style-type: none"> - idToken.type must be <i>eMAID</i> - iso15118CertificateHashData may be provided - certificate is provided <p>* Step 3:</p> <p>Message: TransactionEventRequest</p> <ul style="list-style-type: none"> - triggerReason must be <i>Authorized</i> <p>Post scenario validations:</p> <p>N/a</p> |

TC_C_53_CS: Authorization using Contract Certificates 15118 - Online - Central contract validation fails

| | |
|-------------------|---|
| Test case name | Authorization using Contract Certificates 15118 - Online - Central contract validation fails |
| Test case Id | TC_C_53_CS |
| Use case Id(s) | C07 |
| Requirement(s) | N/a |
| System under test | Charging Station |
| Description | The Charging Station is able to authorize with contract certificates when it supports ISO 15118. |
| Purpose | To verify if the Charging Station is able to handle an invalid contract certificate. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The MO Root certificate that is needed to validate the EV Contract certificate must NOT be installed at the Charging Station. - The Charging Station supports central contract validation. |

| |
|---|
| Before (Preparations) |
| <p>Configuration State:</p> <p>TxCtrlr.TxStartPoint contains one or more of <i>PowerPathClosed</i>, <i>Authorized</i>, <i>EVConnected</i>, <i>ParkingBayOccupancy</i></p> <p>AuthCtrlr.AuthEnabled is <i>true</i> (If implemented AND ReadWrite)</p> <p>AuthCtrlr.DisableRemoteAuthorization is <i>false</i> (If implemented)</p> <p>For the ISO15118Ctrlr of the EVSE involved in the PnC transaction:</p> <p>ISO15118Ctrlr.CentralContractValidationAllowed is <i>true</i></p> <p>ISO15118Ctrlr.PnCEnabled is <i>true</i></p> <p>Memory State:</p> <p><i>CertificateInstalled</i> for certificateType <i>V2GRootCertificate</i></p> <p><i>RenewV2GChargingStationCertificate</i></p> <p>Reusable State(s):</p> <p>State is <i>EVConnectedPreSession</i></p> |

| Main (Test scenario) | |
|--|---|
| Charging Station | CSMS |
| <p>1. The Charging Station sends an AuthorizeRequest.</p> <p><u>Note(s):</u> -The test case should be robust enough to also handle a GetCertificateStatusRequest</p> | <p>2. The Test System responds with a AuthorizeResponse with idTokenInfo.status <i>Invalid</i> and certificateStatus = <i>CertificateRevoked</i></p> |

| |
|--|
| Tool validations |
| <p>* Step 1:</p> <p>Message: AuthorizeRequest</p> <ul style="list-style-type: none"> - idToken.type must be <i>eMAID</i> - iso15118CertificateHashData may be provided - certificate is provided <p>Post scenario validations:</p> <p>EV is not authorized and shall not charge:</p> <p>Charging Station does not send TransactionEventRequest with:</p> <ul style="list-style-type: none"> - triggerReason = <i>Authorized</i> or chargingState = <i>Charging</i> |

TC_C_54_CS: Authorization using Contract Certificates 15118 - Offline - ContractValidationOffline is true

| | |
|-------------------|--|
| Test case name | Authorization using Contract Certificates 15118 - Offline - ContractValidationOffline is true |
| Test case Id | TC_C_54_CS |
| Use case Id(s) | C07 |
| Requirement(s) | C07.FR.08,C07.FR.09,C07.FR.10,C07.FR.11,C07.FR.12 |
| System under test | Charging Station |
| Description | The Charging Station is able to authorize with contract certificates for an EMAID that exists in authorization cache or local authorization list, while offline. |
| Purpose | To verify if the Charging Station is able to authorize using contract certificates, while it is offline. |
| Prerequisite(s) | The Charging Station supports authorization methods other than NoAuthorization |

| |
|--|
| Before (Preparations) |
| <p>Configuration State:</p> <p>TxCtrlr.TxStartPoint contains one or more of <i>PowerPathClosed</i>, <i>Authorized</i>, <i>EVConnected</i>, <i>ParkingBayOccupancy</i></p> <p>AuthCtrlr.AuthEnabled is <i>true</i> (If implemented AND ReadWrite)</p> <p>AuthCacheCtrlr.Enabled is <i>true</i> OR LocalAuthListCtrlr.Enabled is <i>true</i></p> <p>OfflineTxForUnknownIdEnabled is <i>false</i> (If implemented)</p> <p>OfflineThreshold is <i><Configured RetryBackOffWaitMinimum_duration> + 60.0</i></p> <p>RetryBackOffWaitMinimum is <i><Configured RetryBackOffWaitMinimum_duration></i></p> <p>RetryBackOffRandomRange is <i>0</i></p> <p><u>Note:</u></p> <p><i><Configured RetryBackOffWaitMinimum_duration> should be long enough to execute manual tasks></i></p> <p>For ISO15118Ctrlr of EVSE involved in PnC transaction:</p> <p>ISO15118Ctrlr.ContractValidationOffline is <i>true</i></p> <p>ISO15118Ctrlr.PnCEnabled is <i>true</i></p> <p>Memory State:</p> <p><i>CertificateInstalled</i> for certificateType <i>V2GRootCertificate</i></p> <p><i>CertificateInstalled</i> for certificateType <i>MORootCertificate</i></p> <p><i>RenewV2GChargingStationCertificate</i></p> <p><i>IdTokenCached15118</i> for <i><Configured valid ISO 15118 IdToken></i> (If implemented)</p> <p><i>IdTokenLocalAuthList</i> for <i><Configured valid ISO 15118 IdToken></i> (If implemented)</p> <p>Reusable State(s):</p> <p>N/a</p> |

| Main (Test scenario) | |
|--|--|
| Charging Station | CSMS |
| 1. The Test System closes the WebSocket connection AND does not accept a reconnect. | |
| <u>Manual Action</u> : Drive EV into parking bay. | |
| <u>Manual Action</u> : Connect the EV and EVSE. | |
| <u>Notes(s)</u> : The Test System will wait for <Configured Transaction Duration> seconds. | |
| 2. The Test System accepts the reconnection attempt from the Charging Station after <Configured Transaction Duration> seconds. | |
| 3. The Charging Station notifies the CSMS about the status change of the connector. | 4. The Test System responds accordingly. |
| 5. The Charging Station sends a TransactionEventRequest <u>Note(s)</u> : - This step needs to be executed when TxStartPoint contains EVConnected OR the transaction already started (in the case TxStartPoint contains ParkingBayOccupancy) | 6. The Test System responds with a TransactionEventResponse |
| 7. The Charging Station sends a TransactionEventRequest | 8. The Test System responds with a TransactionEventResponse With idTokenInfo.status Accepted |

| |
|--|
| Main (Test scenario) |
| 9. Execute Reusable State <i>EnergyTransferStarted</i> |
| Tool validations |
| <div>* Step 3: Message: StatusNotificationRequest - connectorStatus must be <i>Occupied</i> Message: NotifyEventRequest - eventData[0].trigger must be <i>Delta</i> - eventData[0].actualValue must be <i>Occupied</i> - eventData[0].component.name must be <i>Connector</i> - eventData[0].variable.name must be <i>AvailabilityState</i> * Step 5: Message: TransactionEventRequest - triggerReason must be <i>CablePluggedIn</i> - transactionInfo.chargingState must be <i>EVConnected</i> - offline <i>true</i> * Step 7: Message: TransactionEventRequest - triggerReason must be <i>Authorized</i> - offline <i>true</i></div> |
| Post scenario validations: N/a |

TC_C_55_CS: Authorization using Contract Certificates 15118 - Offline - ContractValidationOffline is false

| | |
|-------------------|---|
| Test case name | Authorization using Contract Certificates 15118 - Offline - ContractValidationOffline is false |
| Test case Id | TC_C_55_CS |
| Use case Id(s) | C07 |
| Requirement(s) | C07.FR.07 |
| System under test | Charging Station |
| Description | The Charging Station will not authorize with contract certificates when offline. |
| Purpose | To verify if the Charging Station is able to handle being offline and not allowing a charging session to start, when it is configured to do so. |
| Prerequisite(s) | The Charging Station supports authorization methods other than NoAuthorization |

| |
|---|
| <p>Before (Preparations)</p> <p>Configuration State: TxCtrlr.TxStartPoint contains one or more of <i>PowerPathClosed</i>, <i>Authorized</i>, <i>EVConnected</i>, <i>ParkingBayOccupancy</i> AuthCtrlr.AuthEnabled is <i>true</i> (If implemented AND ReadWrite) AuthCacheCtrlr.Enabled is <i>true</i> OR LocalAuthListCtrlr.Enabled is <i>true</i> OfflineTxForUnknownIdEnabled is <i>false</i> (If implemented) OfflineThreshold is <i><Configured RetryBackOffWaitMinimum_duration> + 60.0</i> RetryBackOffWaitMinimum is <i><Configured RetryBackOffWaitMinimum_duration></i> RetryBackOffRandomRange is <i>0</i> <u>Note:</u> <i><Configured RetryBackOffWaitMinimum_duration> should be long enough to execute manual tasks></i> For the ISO15118Ctrlr of the EVSE involved in the PnC transaction: ISO15118Ctrlr.ContractValidationOffline is <i>false</i> ISO15118Ctrlr.PnCEnabled is <i>true</i></p> <p>Memory State: <i>CertificateInstalled</i> for certificateType <i>V2GRootCertificate</i> <i>CertificateInstalled</i> for certificateType <i>MORootCertificate</i> <i>RenewV2GChargingStationCertificate</i> <i>IdTokenCached15118</i> for <i><Configured valid ISO 15118 IdToken></i> (If implemented) <i>IdTokenLocalAuthList</i> for <i><Configured valid ISO 15118 IdToken></i> (If implemented)</p> <p>Reusable State(s): N/a</p> |
|---|

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Test System closes the WebSocket connection AND does not accept a reconnect. | |
| <u>Manual Action:</u> Drive EV into parking bay. | |
| <u>Manual Action:</u> Connect the EV and EVSE. | |
| <u>Note(s):</u> The tool will wait for <i><Configured Transaction Duration></i> seconds. | |
| 2. The Test System accepts the reconnection attempt from the Charging Station after <i><Configured Transaction Duration></i> seconds. | |
| 3. The Charging Station notifies the CSMS about the status change of the connector. | 4. The Test System responds accordingly. |
| 5. The Charging Station sends a TransactionEventRequest <u>Note(s):</u> - This step needs to be executed when TxStartPoint contains <i>EVConnected</i> OR the transaction already started (in the case TxStartPoint contains <i>ParkingBayOccupancy</i>) | 6. The Test System responds with a TransactionEventResponse |
| 7. The Charging Station has NOT started charging and does NOT send TransactionEventRequest message(s) with triggerReason <i>Authorized</i> OR <i>ChargingStateChanged</i> . | |

| Tool validations |
|--|
| <p>* Step 3:</p> <p>Message: StatusNotificationRequest</p> <ul style="list-style-type: none">- connectorStatus must be <i>Occupied</i> <p>Message: NotifyEventRequest</p> <ul style="list-style-type: none">- eventData[0].trigger must be <i>Delta</i>- eventData[0].actualValue must be <i>Occupied</i>- eventData[0].component.name must be <i>Connector</i>- eventData[0].variable.name must be <i>AvailabilityState</i> <p>* Step 5:</p> <p>Message: TransactionEventRequest</p> <ul style="list-style-type: none">- triggerReason must be <i>CablePluggedIn</i>- transactionInfo.chargingState must be <i>EVConnected</i>- offline <i>true</i> |
| <p>Post scenario validations:</p> <p>EV is not authorized and shall not charge:</p> <p>Charging Station does not send TransactionEventRequest with:</p> <ul style="list-style-type: none">- triggerReason = <i>Authorized</i> or chargingState = <i>Charging</i> |

TC_C_56_CS: Local start transaction - Authorization Unknown

| | |
|-------------------|--|
| Test case name | Local start transaction - Authorization Unknown |
| Test case Id | TC_C_56_CS |
| Use case Id(s) | C01 |
| Requirement(s) | C01.FR.02 |
| System under test | Charging Station |
| Description | When a Charging Station needs to charge an EV, it needs to authorize the EV Driver first at the CSMS before the charging can be started or stopped. |
| Purpose | To verify whether the Charging Station is able to handle receiving an Unknown idToken. |
| Prerequisite(s) | The Charging Station does NOT have a cable lock, which prevents the EV driver to connect the EV and EVSE before authorization. The Charging Station supports authorization methods other than NoAuthorization |

| |
|---|
| Before (Preparations) |
| Configuration State: AuthCtrlr.AuthEnabled is <i>true</i> (If implemented AND ReadWrite) AuthCtrlr.DisableRemoteAuthorization is <i>false</i> (If implemented) |
| Memory State: N/a |
| Reusable State(s): State is <i>EVConnectedPreSession</i> |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action</u> : Present invalid idToken. | |
| 1. The Charging Station sends an AuthorizeRequest | 2. The Test System responds with an AuthorizeResponse with idTokenInfo.status <i>Unknown</i> |
| <u>Note(s)</u> : - The Charging Station SHALL NOT send a TransactionEventRequest message after the AuthorizeRequest from step 1. - The Test System waits <Configured message timeout> seconds, before ending the testcase. | |

| |
|---|
| Tool validations |
| * Step 1: Message: AuthorizeRequest - idToken.idToken <Configured invalid_idtoken_idtoken> - idToken.type <Configured invalid_idtoken_type> |
| Post scenario validations: N/a |

TC_C_100_CS: Local start transaction - Authorization first - Cable plugin timeout

| | |
|-------------------|---|
| Test case name | Local start transaction - Authorization first - Cable plugin timeout |
| Test case Id | TC_C_100_CS |
| Use case Id(s) | C01 |
| Requirement(s) | C01.FR.26 |
| System under test | Charging Station |
| Description | OCPP 2.x allows an EV driver to either first connect the EV and EVSE OR present a form of identification. Both sequences will result in being able to charge. |
| Purpose | To verify if the Charging Station is able to deauthorize the transaction after the EVConnectionTimeout has expired. |
| Prerequisite(s) | The Charging Station supports at least one authorization method described at the following Use cases; C01, C02, C04, C06. |

| |
|--|
| Before (Preparations) |
| Configuration State: - TxCtrlr.EVConnectionTimeOut is <Configured ev_connection_timeout> - AuthCtrlr.AuthEnabled is true (If implemented AND ReadWrite) - AuthCtrlr.DisableRemoteAuthorization is false (If implemented) - AuthCacheCtrlr.Enabled is false (If implemented) - AuthCtrlr.LocalPreAuthorize is false |
| Memory State: N/a |
| Reusable State(s): State is <i>Authorized</i> (local) |

| Main (Test scenario) | |
|---|---|
| Charging Station | CSMS |
| <p>1. The Charging Station sends a TransactionEventRequest</p> <p><u>Note(s):</u> - This step only needs to be executed after the <Configured ev_connection_timeout> expires, if the transaction has been started. So in the case TxStartPoint contains ParkingBayOccupancy OR Authorized</p> | <p>2. The Test System responds with a TransactionEventResponse</p> |
| <p><u>Note(s):</u> - Optionally the Charging Station can send a StatusNotificationRequest or NotifyEventRequest with status Available</p> | |
| <p><u>Manual Action:</u> Connect the EV and EVSE.</p> | |
| <p><u>Note(s):</u> - This step only needs to be executed after the <Configured EV connection timeout> expires, if the transaction has been started. So in the case TxStartPoint contains ParkingBayOccupancy OR Authorized</p> | |

| |
|---|
| Tool validations |
| * Step 1: Message: TransactionEventRequest - triggerReason must be <i>EVConnectTimeout</i> If <Configured TxStopPoint> contains <i>Authorized</i> then eventType must be <i>Ended</i> AND transactionInfo.stoppedReason must be <i>Timeout</i> Else eventType must be <i>Updated</i> |
| Post scenario validations: - Power transfer will not be started |

TC_C_127_CS: Ad hoc payment via static or dynamic QR code - no URL parameters

| | |
|-------------------|---|
| Test case name | Ad hoc payment via static or dynamic QR code - no URL parameters |
| Test case Id | TC_C_127_CS |
| Use case Id(s) | C25 |
| Requirement(s) | C25.FR.01, C25.FR.26, C25.FR.27, C25.FR.28 |
| System under test | Charging Station |
| Description | To provide a static or dynamic QR code with a URL for ad hoc payment |
| Purpose | To verify if the Charging Station supports ad hoc payments with a url without URL parameters. |
| Prerequisite(s) | Charging Station supports QR code payment |

| |
|---|
| Before (Preparations) |
| <p>Configuration State:</p> <p>TariffCostCtrlr.Enabled[Tariff] <i>true</i></p> <p>TariffCostCtrlr.Enabled[Cost] <i>true</i></p> <p>TariffCostCtrlr.Currency is <i>EUR</i></p> <p>AuthCtrlr.AuthorizeRemoteStart is <i>true</i></p> <p>WebPaymentsCtrlr.Enabled is <i>true</i></p> <p>WebPaymentsCtrlr.URLParameters is absent or empty</p> <p>WebPaymentsCtrlr.TOTPVersion is <i>v1</i></p> <p>WebPaymentsCtrlr.ValidityTime is <i>120</i></p> <p>WebPaymentsCtrlr.SharedSecret is <i>12345678</i></p> <p>WebPaymentsCtrlr.Length is <i>8</i></p> <p>Memory State:</p> <p>N/a</p> <p>Reusable State(s):</p> <p>N/a</p> |

| Main (Test scenario) | |
|--|--|
| Charging Station | CSMS |
| 2. The Charging Station responds with a SetDefaultTariffResponse | 1. The Test System sends a SetDefaultTariffRequest with evseld 0 tariff.tariffId <i>Test System1</i> tariff.currency <i>EUR</i> tariff.fixedFee.taxRates[0].type <i>MyTax</i> tariff.fixedFee.taxRates[0].tax 21 tariff.fixedFee.prices.priceFixed[0] 12.45 |
| <u>Note(s):</u> - The Charging Station displays a dynamic QR code according to configuration variables in <i>WebPaymentsCtrlr</i> | |
| Simulate starting a transaction with an QR code | |
| 4. The Charging Station SHALL respond with a NotifyWebPaymentStartedResponse | 3. The Test System sends a NotifyWebPaymentStartedRequest with evseld <configured evseld> timeout 5 |
| 5. Execute Reusable State <i>AuthorizedPaymentTerminal</i> (Remote) | |
| 6. Execute Reusable State <i>EVConnectedPreSession</i> | |
| 7. Execute Reusable State <i>EnergyTransferStarted</i> | |

| Tool validations |
|---|
| <p>* Step 5 and 6 and 7 combined: (First transactionEventRequest with idToken)</p> <p>Message TransactionEventRequest with:</p> <ul style="list-style-type: none">- idToken must be <i><Not empty></i>- idToken.type must be <i>DirectPayment</i>- transactionInfo.transactionId must be <i><transactionId></i>- transactionInfo.transactionLimit must be omitted |
| <p>Post scenario validations:</p> <p>N/a</p> |

D Local Authorization List Management

TC_D_01_CS: Send Local Authorization List - Full

| | |
|-------------------|---|
| Test case name | Send Local Authorization List - Full |
| Test case Id | TC_D_01_CS |
| Use case Id(s) | D01 |
| Requirement(s) | D01_FR_02, D01_FR_15 |
| System under test | Charging Station |
| Description | The CSMS sends a Local Authorization List which a Charging Station can use for the authorization of idTokens. The list MAY be either a full list to replace the current list in the Charging Station or it MAY be a differential list with updates to be applied to the current list in the Charging Station. |
| Purpose | To verify if the Charging Station is able to replace the Local Authorization List according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | The Charging Station supports the Local Authorization List feature |

| |
|--|
| Before (Preparations) |
| Configuration State: LocalAuthListEnabled is <i>true</i> (If implemented) *Configured versionNumber > 0 |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a SendLocalListResponse | 1. The Test System sends a SendLocalListRequest with - updateType <i>Full</i> - versionNumber <Configured <i>versionNumber</i> > - localAuthorizationList[0].idToken.idToken <Configured <i>valid_idtoken_idtoken</i> > - localAuthorizationList[0].idToken.type <Configured <i>valid_idtoken_type</i> > |
| 4. The Charging Station responds with a GetLocalListVersionResponse | 3. The Test System sends a GetLocalListVersionRequest |

| |
|---|
| Tool validations |
| * Step 2: Message SendLocalListResponse - status <i>Accepted</i> * Step 4: Message GetLocalListVersionResponse - versionNumber <Equal to version sent in step 1> |
| Post scenario validations: - N/a |

TC_D_02_CS: Send Local Authorization List - Differential Update

| | |
|-------------------|---|
| Test case name | Send Local Authorization List - Differential Update |
| Test case Id | TC_D_02_CS |
| Use case Id(s) | D01 |
| Requirement(s) | D01_FR_02, D01_FR_16 |
| System under test | Charging Station |
| Description | The CSMS sends a Local Authorization List which a Charging Station can use for the authorization of idTokens. The list MAY be either a full list to replace the current list in the Charging Station or it MAY be a differential list with updates to be applied to the current list in the Charging Station. |
| Purpose | To verify if the Charging Station is able to replace the Local Authorization List in differential type according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | The Charging Station supports the Local Authorization List feature |

| |
|---|
| Before (Preparations) |
| Configuration State: LocalAuthListEnabled is <i>true</i> (If implemented) *Configured versionNumber > 0 |
| Memory State: <i>IdTokenLocalAuthList</i> for <Configured valid idtoken fields> |
| Reusable State: N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a SendLocalListResponse | 1. The Test System sends a SendLocalListRequest with - updateType <i>Differential</i> - versionNumber <Configured versionNumber + 1> - localAuthorizationList[0].idToken.idToken <Configured <i>valid_idtoken_idtoken2</i> > - localAuthorizationList[0].idToken.type <Configured <i>valid_idtoken_type2</i> > |
| <u>Note(s):</u> The message send by Test System is within <i>ItemsPerMessageSendLocalList</i> AND <i>BytesPerMessageSendLocalList</i> range. | |
| 4. The Charging Station responds with a GetLocalListVersionResponse | 3. The Test System sends a GetLocalListVersionRequest |

| |
|---|
| Tool validations |
| * Step 2: Message SendLocalListResponse - status <i>Accepted</i> * Step 4: Message GetLocalListVersionResponse - versionNumber <Equal to version send in step 1> |
| Post scenario validations: - N/a |

TC_D_03_CS: Send Local Authorization List - Differential Remove

| | |
|-------------------|---|
| Test case name | Send Local Authorization List - Differential Remove |
| Test case Id | TC_D_03_CS |
| Use case Id(s) | D01 |
| Requirement(s) | D01_FR_02, D01_FR_17 |
| System under test | Charging Station |
| Description | The CSMS sends a Local Authorization List which a Charging Station can use for the authorization of idTokens. The list MAY be either a full list to replace the current list in the Charging Station or it MAY be a differential list with updates to be applied to the current list in the Charging Station. |
| Purpose | To verify if the Charging Station is able to remove items from the Local Authorization List when send in differential type with data without idToken according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | The Charging Station supports the Local Authorization List feature |

| |
|---|
| Before (Preparations) |
| Configuration State: LocalAuthListEnabled is <i>true</i> (If implemented) *Configured versionNumber > 0 |
| Memory State: <i>IdTokenLocalAuthList</i> for <Configured valid idtoken fields> |
| Reusable State: N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a SendLocalListResponse | 1. The Test System sends a SendLocalListRequest with - updateType <i>Differential</i> - versionNumber <Configured versionNumber + 1> - localAuthorizationList <Contains <i>AuthorizationData</i> elements without <i>idTokenInfo</i> > |
| <u>Note(s)</u> : The message send by Test System is within <i>ItemsPerMessageSendLocalList</i> AND <i>BytesPerMessageSendLocalList</i> range. | |
| 4. The Charging Station responds with a GetLocalListVersionResponse | 3. The Test System sends a GetLocalListVersionRequest |

| |
|---|
| Tool validations |
| * Step 2: Message SendLocalListResponse - status <i>Accepted</i> * Step 3: Message GetLocalListVersionResponse - versionNumber <Equal to version sent in step 1> |
| Post scenario validations: - N/a |

TC_D_04_CS: Send Local Authorization List - Full with empty list

| | |
|-------------------|---|
| Test case name | Send Local Authorization List - Full with empty list |
| Test case Id | TC_D_04_CS |
| Use case Id(s) | D01 |
| Requirement(s) | D01_FR_02, D01_FR_04 |
| System under test | Charging Station |
| Description | The CSMS sends a Local Authorization List which a Charging Station can use for the authorization of idTokens. The list MAY be either a full list to replace the current list in the Charging Station or it MAY be a differential list with updates to be applied to the current list in the Charging Station. |
| Purpose | To verify if the Charging Station is able to remove all items from the Local Authorization List when send in full type with no data according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | The Charging Station supports the Local Authorization List feature |

| |
|---|
| Before (Preparations) |
| Configuration State: LocalAuthListEnabled is <i>true</i> (If implemented) *Configured <i>versionNumber</i> > 0 |
| Memory State: <i>IdTokenLocalAuthList</i> for <Configured valid idtoken fields> |
| Reusable State: N/a |

| Main (Test scenario) | |
|--|--|
| Charging Station | CSMS |
| 2. The Charging Station responds with a SendLocalListResponse | 1. The Test System sends a SendLocalListRequest with - updateType <i>Full</i> - versionNumber <Configured <i>versionNumber</i> > - localAuthorizationList <Empty> |
| 4. The Charging Station responds with a GetLocalListVersionResponse | 3. The Test System sends a GetLocalListVersionRequest |

| |
|--|
| Tool validations |
| * Step 2: Message SendLocalListResponse - status <i>Accepted</i> * Step 4: Message GetLocalListVersionResponse - versionNumber <Configured <i>versionNumber</i> > |
| Post scenario validations: - N/a |

TC_D_05_CS: Send Local Authorization List - Differential with empty list

| | |
|-------------------|---|
| Test case name | Send Local Authorization List - Differential with empty list |
| Test case Id | TC_D_05_CS |
| Use case Id(s) | D01 |
| Requirement(s) | D01_FR_02, D01_FR_05 |
| System under test | Charging Station |
| Description | The CSMS sends a Local Authorization List which a Charging Station can use for the authorization of idTokens. The list MAY be either a full list to replace the current list in the Charging Station or it MAY be a differential list with updates to be applied to the current list in the Charging Station. |
| Purpose | To verify if the Charging Station is able to correctly respond on a Local Authorization List when send in differential type with no data according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | The Charging Station supports the Local Authorization List feature |

| |
|--|
| Before (Preparations) |
| Configuration State: LocalAuthListEnabled is <i>true</i> (If implemented) *Configured versionNumber > 0 |
| Memory State: <i>IdTokenLocalAuthList</i> for <Configured valid idtoken fields> |
| Reusable State: N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a SendLocalListResponse | 1. The Test System sends a SendLocalListRequest with - updateType <i>Differential</i> - versionNumber <Configured versionNumber + 1> - localAuthorizationList <Empty> |
| 4. The Charging Station responds with a GetLocalListVersionResponse | 3. The Test System sends a GetLocalListVersionRequest |

| |
|---|
| Tool validations |
| * Step 2: Message SendLocalListResponse - status <i>Accepted</i> * Step 4: Message GetLocalListVersionResponse - versionNumber <Equal to the version send in step 1> |
| Post scenario validations: - N/a |

TC_D_06_CS: Send Local Authorization List - VersionMismatch

| | |
|-------------------|---|
| Test case name | Send Local Authorization List - VersionMismatch |
| Test case Id | TC_D_06_CS |
| Use case Id(s) | D01 |
| Requirement(s) | D01_FR_19 |
| System under test | Charging Station |
| Description | The CSMS sends a Local Authorization List which a Charging Station can use for the authorization of idTokens. The list MAY be either a full list to replace the current list in the Charging Station or it MAY be a differential list with updates to be applied to the current list in the Charging Station. |
| Purpose | To verify if the Charging Station is able to correctly respond on a Local Authorization List when send in differential type with with a faulty version number according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | The Charging Station supports the Local Authorization List feature |

| |
|---|
| Before (Preparations) |
| Configuration State: LocalAuthListEnabled is <i>true</i> (If implemented) *Configured versionNumber > 1 |
| Memory State: <i>IdTokenLocalAuthList</i> for <Configured valid idtoken fields> |
| Reusable State: N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a SendLocalListResponse | 1. The Test System sends a SendLocalListRequest with - updateType <i>Differential</i> - versionNumber <Configured <i>versionNumber</i> - 1> - localAuthorizationList <Not Empty> |
| 4. The Charging Station responds with a GetLocalListVersionResponse | 3. The Test System sends a GetLocalListVersionRequest |

| |
|---|
| Tool validations |
| * Step 2: Message SendLocalListResponse - status <i>VersionMismatch</i> * Step 4: Message GetLocalListVersionResponse - versionNumber <Configured <i>versionNumber</i> > |
| Post scenario validations: - N/a |

TC_D_07_CS: Send Local Authorization List - Persistent over reboot

| | |
|-------------------|---|
| Test case name | Send Local Authorization List - Persistent over reboot |
| Test case Id | TC_D_07_CS |
| Use case Id(s) | D01 |
| Requirement(s) | D01_FR_10 |
| System under test | Charging Station |
| Description | The CSMS sends a Local Authorization List which a Charging Station can use for the authorization of idTokens. The list MAY be either a full list to replace the current list in the Charging Station or it MAY be a differential list with updates to be applied to the current list in the Charging Station. |
| Purpose | To verify if the Charging Station is able to save the Local Authorization List persistent over reboot according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | The Charging Station supports the Local Authorization List feature and stores it in non-volatile memory |

| |
|--|
| Before (Preparations) |
| Configuration State: LocalAuthListEnabled is <i>true</i> (If implemented) *Configured versionNumber > 0 |
| Memory State: <i>IdTokenLocalAuthList</i> for <Configured valid idtoken fields> |
| Reusable State: <i>Booted</i> |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a GetLocalListVersionResponse | 1. The Test System sends a GetLocalListVersionRequest |

| |
|--|
| Tool validations |
| * Step 2: Message GetLocalListVersionResponse - versionNumber <Configured versionNumber> |
| Post scenario validations: - N/a |

TC_D_08_CS: Get Local List Version - Success

| | |
|-------------------|---|
| Test case name | Get Local List Version - Success |
| Test case Id | TC_D_08_CS |
| Use case Id(s) | D02 |
| Requirement(s) | D02_FR_01 |
| System under test | Charging Station |
| Description | The CSMS can request a Charging Station for the version number of the Local Authorization List by sending a GetLocalListVersionRequest. |
| Purpose | To verify if the Charging Station is able to respond the Local Authorization List version number according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | The Charging Station supports the Local Authorization List feature |

| |
|--|
| Before (Preparations) |
| Configuration State: LocalAuthListEnabled is <i>true</i> (If implemented) *Configured versionNumber > 0 |
| Memory State: <i>IdTokenLocalAuthList</i> for <Configured valid idtoken fields> |
| Reusable State: N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a GetLocalListVersionResponse | 1. The Test System sends a GetLocalListVersionRequest |

| |
|--|
| Tool validations |
| * Step 2: Message GetLocalListVersionResponse - versionNumber <Configured versionNumber> |
| Post scenario validations: - N/a |

TC_D_10_CS: Get Local List Version - Function disabled

| | |
|--------------------------|--|
| Test case name | Get Local List Version - Function disabled |
| Test case Id | TC_D_10_CS |
| Use case Id(s) | D02 |
| Requirement(s) | D02_FR_03 |
| System under test | Charging Station |
| Description | The CSMS can request a Charging Station for the version number of the Local Authorization List by sending a GetLocalListVersionRequest . |
| Purpose | To verify if the Charging Station is able to respond the Local Authorization List version number according to the mechanism as described in the OCPP specification when the LocalAuthListEnabled is set to false. |
| Prerequisite(s) | The Charging Station supports the Local Authorization List feature |

| |
|---|
| Before (Preparations) |
| Configuration State: LocalAuthListEnabled is <i>false</i> (If implemented) |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a GetLocalListVersionResponse | 1. The Test System sends a GetLocalListVersionRequest |

| |
|---|
| Tool validations |
| * Step 2: Message GetLocalListVersionResponse - versionNumber 0 |
| Post scenario validations: - N/a |

E Transactions

TC_E_01_CS: Start transaction options - PowerPathClosed

| | |
|-------------------|--|
| Test case name | Start transaction options - PowerPathClosed |
| Test case Id | TC_E_01_CS |
| Use case Id(s) | E01(S5) |
| Requirement(s) | E01.FR.05, E01.FR.07, E01.FR.10, E01.FR.15, E01.FR.16 |
| System under test | Charging Station |
| Description | OCPP 2.x.x has a flexible transaction mechanism that allows the transaction start and stop points to be configured differently. This test covers one of the start options. |
| Purpose | To verify if the Charging Station starts a transaction when the power path has been closed and it has been configured to do so. |
| Prerequisite(s) | - The Charging Station does NOT have the following configuration; The mutability of TxStartPoint is <i>ReadOnly</i> AND (the value <i>PowerPathClosed</i> is NOT set OR (<i>ParkingBayOccupancy</i> OR <i>EVConnected</i> OR <i>Authorized</i> OR <i>DataSigned</i>), is set). - If the mutability of TxStartPoint is <i>ReadWrite</i> , then the value <i>PowerPathClosed</i> must be supported. |

| |
|--|
| Before (Preparations) |
| Configuration State: If the mutability of TxStartPoint is <i>ReadWrite</i> then TxStartPoint contains <i>PowerPathClosed</i> |
| Memory State: N/a |
| Reusable State(s): State is <i>Authorized</i> |

| | |
|--|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>EVConnectedPreSession</i> | |
| 2. Execute Reusable State <i>EnergyTransferStarted</i> | |

| |
|--|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_E_02_CS: Start transaction options - EnergyTransfer

| | |
|-------------------|--|
| Test case name | Start transaction options - EnergyTransfer |
| Test case Id | TC_E_02_CS |
| Use case Id(s) | E01(S6) |
| Requirement(s) | E01.FR.06, E01.FR.07, E01.FR.10, E01.FR.15, E01.FR.16 |
| System under test | Charging Station |
| Description | OCPP 2.x.x has a flexible transaction mechanism that allows the transaction start and stop points to be configured differently. This test covers one of the start options. |
| Purpose | To verify if the Charging Station starts a transaction when the energy transfer starts and it has been configured to do so. |
| Prerequisite(s) | <p>- The Charging Station does NOT have the following configuration; The mutability of TxStartPoint is <i>ReadOnly</i> AND (the value <i>EnergyTransfer</i> is NOT set OR (<i>ParkingBayOccupancy</i> OR <i>EVConnected</i> OR <i>Authorized</i> OR <i>DataSigned</i> OR <i>PowerPathClosed</i>), is set).</p> <p>- If the mutability of TxStartPoint is <i>ReadWrite</i>, then the value <i>EnergyTransfer</i> must be supported.</p> |

Before (Preparations)

Configuration State:

If the mutability of **TxStartPoint** is *ReadWrite* then **TxStartPoint** contains *EnergyTransfer*

Memory State:

N/a

Reusable State(s):

State is *Authorized*

Main (Test scenario)

| | |
|---|--|
| Charging Station | CSMS |
| <u>Manual Action:</u> Connect the EV and EVSE. | |
| 1. The Charging Station notifies the CSMS about the status change of the connector. | 2. The Test System responds accordingly. |
| 3. The Charging Station sends a TransactionEventRequest | 4. The Test System responds with a TransactionEventResponse |

Tool validations

* Step 1:

Message: **StatusNotificationRequest**

- **connectorStatus** must be *Occupied*

Message: **NotifyEventRequest**

- **eventData[0].trigger** must be *Delta*

- **eventData[0].actualValue** must be *Occupied*

- **eventData[0].component.name** must be *Connector*

- **eventData[0].variable.name** must be *AvailabilityState*

* Step 3:

Message: **TransactionEventRequest**

- **eventType** must be *Started*

- If the Test System is configured to start transactions using a *RequestStartTransactionRequest* message then

triggerReason must be *RemoteStart*

Else **triggerReason** must be *ChargingStateChanged* or *Authorized*

- **idToken.idToken** <Configured valid_idtoken_idtoken>

- **idToken.type** <Configured valid_idtoken_type>

- **evse** must be provided

- **evse.connectorId** must be provided

- **transactionInfo.chargingState** must be *Charging*

Post scenario validations:

N/a

TC_E_09_CS: Start transaction options - EVConnected

| | |
|-------------------|--|
| Test case name | Start transaction options - EVConnected |
| Test case Id | TC_E_09_CS |
| Use case Id(s) | E01(S2) |
| Requirement(s) | E01.FR.02, E01.FR.07, E01.FR.10, E01.FR.15, E01.FR.16 |
| System under test | Charging Station |
| Description | OCPP 2.x.x has a flexible transaction mechanism that allows the transaction start and stop points to be configured differently. This test covers one of the start options. |
| Purpose | To verify if the Charging Station starts a transaction when the EV and EVSE are connected and it has been configured to do so. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The Charging Station does NOT have the following configuration; The mutability of TxStartPoint is <i>ReadOnly</i> AND (the value <i>EVConnected</i> is NOT set OR <i>ParkingBayOccupancy</i> is set). - If the mutability of TxStartPoint is <i>ReadWrite</i>, then the value <i>EVConnected</i> must be supported. |

| |
|--|
| Before (Preparations) |
| Configuration State: If the mutability of TxStartPoint is <i>ReadWrite</i> then TxStartPoint contains <i>EVConnected</i> |
| Memory State: N/a |
| Reusable State(s): State is <i>ParkingBayOccupied</i> |

| | |
|--|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>EVConnectedPreSession</i> | |
| 2. Execute Reusable State <i>Authorized</i> | |
| 3. Execute Reusable State <i>EnergyTransferStarted</i> | |

| |
|--|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_E_10_CS: Start transaction options - Authorized - Local

| | |
|-------------------|--|
| Test case name | Start transaction options - Authorized - Local |
| Test case Id | TC_E_10_CS |
| Use case Id(s) | E01(S3) AND (C01 OR C02 OR C04 OR C06) |
| Requirement(s) | E01.FR.03, E01.FR.07, E01.FR.10, E01.FR.15, E01.FR.16 AND C01.FR.02, C02.FR.01, C06.FR.02 |
| System under test | Charging Station |
| Description | OCPP 2.x.x has a flexible transaction mechanism that allows the transaction start and stop points to be configured differently. This test covers one of the start options. |
| Purpose | To verify if the Charging Station starts a transaction when the EV and EVSE are connected and it has been configured to do so. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The Charging Station supports at least one authorization method described at the following Use cases; C01, C02, C04, C06. - The Charging Station does NOT have the following configuration; The mutability of TxStartPoint is <i>ReadOnly</i> AND the value <i>Authorized</i> is NOT set. - If the mutability of TxStartPoint is <i>ReadWrite</i>, then the value <i>Authorized</i> must be supported. |

| |
|---|
| Before (Preparations) |
| Configuration State: If the mutability of TxStartPoint is <i>ReadWrite</i> then TxStartPoint contains <i>Authorized</i> AuthCtrlr.AuthEnabled is <i>true</i> (If implemented AND <i>ReadWrite</i>) AuthCtrlr.DisableRemoteAuthorization is <i>false</i> (If implemented) |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State Authorized (Local) | |
| 2. Execute Reusable State EVConnectedPreSession | |
| 3. Execute Reusable State EnergyTransferStarted | |

| |
|--|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_E_13_CS: Start transaction options - Authorized - Remote

| | |
|-------------------|---|
| Test case name | Start transaction options - Authorized - Remote |
| Test case Id | TC_E_13_CS |
| Use case Id(s) | E01(S3) AND F02 |
| Requirement(s) | E01.FR.03 AND F01.FR.03, F01.FR.04, F01.FR.06, F01.FR.19, F02.FR.01 |
| System under test | Charging Station |
| Description | OCPP 2.x.x has a flexible transaction mechanism that allows the transaction start and stop points to be configured differently. This test covers one of the start options. |
| Purpose | To verify if the Charging Station starts a transaction when the EV and EVSE are connected and it has been configured to do so. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The Charging Station does NOT have the following configuration; The mutability of TxStartPoint is <i>ReadOnly</i> AND the value <i>Authorized</i> is NOT set. - If the mutability of TxStartPoint is <i>ReadWrite</i>, then the value <i>Authorized</i> must be supported. |

Before (Preparations)

Configuration State:

If the mutability of **TxStartPoint** is *ReadWrite* then **TxStartPoint** contains *Authorized*

AuthCtrlr.AuthEnabled is *true* (If implemented AND *ReadWrite*)

AuthCtrlr.DisableRemoteAuthorization is *false* (If implemented)

Memory State:

N/a

Reusable State(s):

N/a

Main (Test scenario)

| | |
|--|------|
| Charging Station | CSMS |
| 1. Execute Reusable State <i>Authorized</i> (Remote) | |
| 2. Execute Reusable State <i>EVConnectedPreSession</i> | |
| 3. Execute Reusable State <i>EnergyTransferStarted</i> | |

Tool validations

N/a

Post scenario validations:

N/a

TC_E_11_CS: Start transaction options - DataSigned

| | |
|-------------------|---|
| Test case name | Start transaction options - DataSigned |
| Test case Id | TC_E_11_CS |
| Use case Id(s) | E01(S4) |
| Requirement(s) | E01.FR.04, E01.FR.07, E01.FR.10, E01.FR.15, E01.FR.16 |
| System under test | Charging Station |
| Description | OCPP 2.x.x has a flexible transaction mechanism that allows the transaction start and stop points to be configured differently. This test covers one of the start options. |
| Purpose | To verify if the Charging Station starts a transaction when the EV and EVSE are connected and it has been configured to do so. |
| Prerequisite(s) | <p>- The Charging Station does NOT have the following configuration; The mutability of TxStartPoint is <i>ReadOnly</i> AND (the value <i>DataSigned</i> is NOT set OR (<i>ParkingBayOccupancy</i> OR <i>EVConnected</i> OR <i>Authorized</i>), is set).</p> <p>- If the mutability of TxStartPoint is <i>ReadWrite</i>, then the value <i>DataSigned</i> must be supported.</p> |

| |
|---|
| Before (Preparations) |
| <p>Configuration State:</p> <p>If the mutability of TxStartPoint is <i>ReadWrite</i> then TxStartPoint contains <i>DataSigned</i> SampledDataCtrlr.SignReadings is <i>true</i></p> |
| <p>Memory State:</p> <p>N/a</p> |
| <p>Reusable State(s):</p> <p>State is <i>Authorized</i></p> |

| Main (Test scenario) | |
|---|--|
| Charging Station | CSMS |
| <u>Manual Action</u> : Connect the EV and EVSE. | |
| 1. The Charging Station notifies the CSMS about the status change of the connector. | 2. The Test System responds accordingly. |
| 3. The Charging Station sends a TransactionEventRequest | 4. The Test System responds with a TransactionEventResponse |
| 5. The Charging Station sends a TransactionEventRequest | 6. The Test System responds with a TransactionEventResponse |

| Tool validations |
|--|
| <p>* Step 1:</p> <p>Message: StatusNotificationRequest</p> <ul style="list-style-type: none"> - connectorStatus must be <i>Occupied</i> <p>Message: NotifyEventRequest</p> <ul style="list-style-type: none"> - eventData[0].trigger must be <i>Delta</i> - eventData[0].actualValue must be <i>Occupied</i> - eventData[0].component.name must be <i>Connector</i> - eventData[0].variable.name must be <i>AvailabilityState</i> <p>* Step 3:</p> <p>Message: TransactionEventRequest</p> <ul style="list-style-type: none"> - eventType must be <i>Started</i> - If the Test System is configured to start transactions using a RequestStartTransactionRequest message then triggerReason must be <i>RemoteStart</i> or <i>SignedDataReceived</i> Else triggerReason must be <i>SignedDataReceived</i> - idToken.idToken <Configured valid_idtoken_idtoken> - idToken.type <Configured valid_idtoken_type> - evse must be provided - evse.connectorId must be provided - meterValue is provided with the following values: sampledValue.context is <i>Transaction.Begin</i> sampledValue.signedMeterValue.encodingMethod is not omitted sampledValue.signedMeterValue.publicKey is not omitted sampledValue.signedMeterValue.signedMeterData is not omitted sampledValue.signedMeterValue.signingMethod is not omitted <p>* Step 5:</p> <p>Message: TransactionEventRequest</p> <ul style="list-style-type: none"> - eventType must be <i>Updated</i> - triggerReason must be <i>ChargingStateChanged</i> - transactionInfo.chargingState must be <i>Charging</i> |
| <p>Post scenario validations:</p> <p>N/a</p> |

TC_E_12_CS: Start transaction options - ParkingBayOccupied

| | |
|-------------------|---|
| Test case name | Start transaction options - ParkingBayOccupied |
| Test case Id | TC_E_12_CS |
| Use case Id(s) | E01(S1) |
| Requirement(s) | E01.FR.01, E01.FR.07, E01.FR.10, E01.FR.15, E01.FR.16 |
| System under test | Charging Station |
| Description | OCPP 2.x.x has a flexible transaction mechanism that allows the transaction start and stop points to be configured differently. This test covers one of the start options. |
| Purpose | To verify if the Charging Station starts a transaction when the EV and EVSE are connected and it has been configured to do so. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The Charging Station does NOT have the following configuration; The mutability of TxStartPoint is <i>ReadOnly</i> AND the value <i>ParkingBayOccupancy</i> is NOT set. - If the mutability of TxStartPoint is <i>ReadWrite</i>, then the value <i>ParkingBayOccupancy</i> must be supported. |

| |
|--|
| Before (Preparations) |
| Configuration State: If the mutability of TxStartPoint is <i>ReadWrite</i> then TxStartPoint contains <i>ParkingBayOccupancy</i> |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>ParkingBayOccupied</i> | |
| 2. Execute Reusable State <i>Authorized</i> | |
| 3. Execute Reusable State <i>EVConnectedPreSession</i> | |
| 4. Execute Reusable State <i>EnergyTransferStarted</i> | |

| |
|--|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_E_16_CS: Stop transaction options - Deauthorized - Invalid idToken

| | |
|-------------------|--|
| Test case name | Stop transaction options - Deauthorized - Invalid idToken |
| Test case Id | TC_E_16_CS |
| Use case Id(s) | E06(S3) |
| Requirement(s) | E06.FR.04, E06.FR.15, C15.FR.04 |
| System under test | Charging Station |
| Description | OCPP 2.x.x has a flexible transaction mechanism that allows the transaction start and stop points to be configured differently. This test covers one of the start options. |
| Purpose | To verify if the Charging Station stops a transaction when the transaction gets deauthorized by the status from the idTokenInfo at a TransactionEventResponse message and it has been configured to do so. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The Charging Station does NOT have the following configuration; The mutability of TxStopPoint is <i>ReadOnly</i> AND the value <i>Authorized</i> OR <i>PowerPathClosed</i> is NOT set OR (<i>EnergyTransfer</i> OR <i>DataSigned</i> is set). - If the mutability of TxStopPoint is <i>ReadWrite</i>, then the value <i>Authorized</i> OR <i>PowerPathClosed</i> must be supported. - The Charging Station supports local start/stop transaction. - The Charging Station supports authorization methods other than <i>NoAuthorization</i> |

Before (Preparations)

Configuration State:

If the mutability of **TxStopPoint** is *ReadWrite* then **TxStopPoint** contains *PowerPathClosed* AND/OR *Authorized*

AuthCtrlr.AuthEnabled is *true* (If implemented AND *ReadWrite*)

AuthCtrlr.DisableRemoteAuthorization is *false* (If implemented)

OfflineTxForUnknownIdEnabled is *true* (If implemented)

StopTxOnInvalidId is *true*

Memory State:

IdTokenCached for <Configured valid idtoken fields> (If implemented)

IdTokenLocalAuthList for <Configured valid idtoken fields> (If implemented)

Reusable State(s):

State is *StartOfflineTransaction*

Main (Test scenario)

| Charging Station | CSMS |
|---|--|
| 1. The Charging Stations sends a TransactionEventRequest <u>Note(s):</u> - The Charging Station will empty its Transaction message queue. This will contain one or more TransactionEventRequest messages | 2. The Test System responds with a TransactionEventResponse <u>Note(s):</u> - The Test System will respond to the TransactionEventRequest containing the idToken, with idtokenInfo.status <i>Invalid</i> |
| 3. The Charging Stations sends a TransactionEventRequest <u>Note(s):</u> - After having emptied its queue, the Charging Station will send a TransactionEventRequest in which it reports it deauthorizes the transaction. | 4. The Test System responds with a TransactionEventResponse |

| Tool validations |
|---|
| <div>* Step 1: Message: TransactionEventRequest - offline must be <i>true</i></div> <div>* Step 3: Message: TransactionEventRequest - eventType must be <i>Ended</i> - triggerReason must be <i>Deauthorized</i> - transactionInfo.stoppedReason is <i>DeAuthorized</i></div> |
| <div>Post scenario validations: N/a</div> |

TC_E_17_CS: Stop transaction options - Deauthorized - EV side disconnect

| | |
|-------------------|--|
| Test case name | Stop transaction options - Deauthorized - EV side disconnect |
| Test case Id | TC_E_17_CS |
| Use case Id(s) | E06(S3) |
| Requirement(s) | E06.FR.04, E06.FR.15 |
| System under test | Charging Station |
| Description | OCPP 2.x.x has a flexible transaction mechanism that allows the transaction start and stop points to be configured differently. This test covers one of the start options. |
| Purpose | To verify if the Charging Station stops a transaction when the transaction gets deauthorized by a connection loss from the EV side and it has been configured to do so. |
| Prerequisite(s) | This testcase is applicable if the value Authorized is a supported value for TxStopPoint AND EVConnected, PowerPathClosed and EnergyTransfer must not be set as TxStopPoint AND StopTxOnEVSideDisconnect true must be a supported value. |

Before (Preparations)

Configuration State:

If the mutability of **TxStopPoint** is *ReadWrite* then **TxStopPoint** contains *Authorized*

StopTxOnEVSideDisconnect is *true*

UnlockOnEVSideDisconnect is *false*

AuthCtrlr.AuthEnabled is *true* (If implemented AND *ReadWrite*)

AuthCtrlr.DisableRemoteAuthorization is *false* (If implemented)

Memory State:

N/a

Reusable State(s):

State is *EnergyTransferSuspended*

Main (Test scenario)

| | |
|---|--|
| Charging Station | CSMS |
| <u>Manual Action:</u> Disconnect the EV and EVSE on EV side (EVSE loses connection with EV). | |
| 1. The Charging Station sends a TransactionEventRequest | 2. The Test System responds with a TransactionEventResponse |
| <u>Manual Action:</u> Present the <i>IdToken</i> that was used to start the transaction. | |
| <u>Note(s):</u> - This manual action needs to be executed when the Charging Station has a detachable cable on the Charging Station side AND <i>UnlockOnEVSideDisconnect</i> is set to <i>false</i> . | |
| <u>Manual Action:</u> Disconnect the EV and EVSE on Charging Station side. | |
| <u>Note(s):</u> - This manual action needs to be executed when the Charging Station has a detachable cable on the Charging Station side. | |
| 3. The Charging Station notifies the CSMS about the current state of the connector. | 4. The Test System responds accordingly. |

| Tool validations |
|---|
| <div>* Step 1: Message: TransactionEventRequest<ul style="list-style-type: none">- triggerReason must be <i>EVCommunicationLost</i>- transactionInfo.chargingState must be <i>Idle</i>- transactionInfo.stoppedReason must be <i>EVDisconnected</i>- eventType must be <i>Ended</i></div> <div>* Step 3: Message: StatusNotificationRequest<ul style="list-style-type: none">- connectorStatus <i>Available</i></div> <div>Message: NotifyEventRequest<ul style="list-style-type: none">- eventData[0].trigger <i>Delta</i>- eventData[0].actualValue <i>"Available"</i>- eventData[0].component.name <i>"Connector"</i>- eventData[0].variable.name <i>"AvailabilityState"</i></div> |
| <div>Post scenario validations: N/a</div> |

TC_E_39_CS: Stop transaction options - Deauthorized - timeout

| | |
|-------------------|---|
| Test case name | Stop transaction options - Deauthorized - timeout |
| Test case Id | TC_E_39_CS |
| Use case Id(s) | E06(S3) |
| Requirement(s) | E06.FR.04, E06.FR.15 |
| System under test | Charging Station |
| Description | OCPP 2.x.x has a flexible transaction mechanism that allows the transaction start and stop points to be configured differently. This test covers one of the start options. |
| Purpose | To verify if the Charging Station stops a transaction when the transaction gets deauthorized because the cable was not plugged in within the Configured durationout and it has been configured to do so. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The Charging Station does NOT have the following configuration; The mutability of TxStopPoint is <i>ReadOnly</i> AND the value <i>Authorized</i> is NOT set. - If the mutability of TxStopPoint is <i>ReadWrite</i>, then the value <i>Authorized</i> must be supported. |

Before (Preparations)

Configuration State:

If the mutability of **TxStopPoint** is *ReadWrite* then **TxStopPoint** contains *Authorized*

- **TxCtrlr.EVConnectionTimeOut** is *<Configured ev_connection_timeout>*

- **AuthCtrlr.AuthEnabled** is *true* (If implemented AND *ReadWrite*)

AuthCtrlr.DisableRemoteAuthorization is *false* (If implemented)

Memory State:

N/a

Reusable State(s):

State is *Authorized*

Main (Test scenario)

| Charging Station | CSMS |
|--|---|
| 1. The Charging Station sends a TransactionEventRequest <u>Note(s):</u> - This step needs to be executed after the <i><Configured ev_connection_timeout></i> expires, if the transaction has been started. So in the case TxStartPoint contains <i>ParkingBayOccupancy</i> OR <i>Authorized</i> | 2. The Test System responds with a TransactionEventResponse |
| <u>Note(s):</u> Step 1 and 2 are optional and will only be expected when the <i>TxStartPoint</i> is set to <i>ParkingBayOccupancy</i> or <i>Authorized</i> . Optionally the Charging Station can send a <i>StatusNotificationRequest</i> or <i>NotifyEventRequest</i> with status <i>Available</i> . | |
| <u>Manual Action:</u> Connect the EV and EVSE on EV side. | |
| <u>Manual Action:</u> Connect the EV and EVSE on EVSE side. | |
| 3. The Charging Station notifies the CSMS about the status change of the connector. | 4. The Test System responds accordingly. |
| 5. The Charging Station sends a TransactionEventRequest <u>Note(s):</u> - This step needs to be executed after the <i><Configured ev_connection_timeout></i> expires, if the transaction has been started. So in the case TxStartPoint contains <i>ParkingBayOccupancy</i> OR <i>Authorized</i> | 6. The Test System responds with a TransactionEventResponse |
| <u>Note(s):</u> Charging Station is allowed to sent a <i>TransactionEventRequest</i> for the <i>cableplugin</i> event when this is applicable, but should not start charging. | |

| Tool validations |
|---|
| <p>* Step 1:</p> <p>Message: TransactionEventRequest</p> <ul style="list-style-type: none">- triggerReason must be <i>EVConnectTimeout</i>- eventType must be <i>Updated</i> if TxStartPoint is <i>ParkingBayOccupancy</i>, else <i>Ended</i>- transactionInfo.stoppedReason must be <i>Timeout</i> <p>* Step 3:</p> <p>Message: StatusNotificationRequest</p> <ul style="list-style-type: none">- connectorStatus must be <i>Occupied</i> <p>Message: NotifyEventRequest</p> <ul style="list-style-type: none">- eventData[0].trigger must be <i>Delta</i>- eventData[0].actualValue must be <i>Occupied</i>- eventData[0].component.name must be <i>Connector</i>- eventData[0].variable.name must be <i>AvailabilityState</i> <p>* Step 5:</p> <p>Message: TransactionEventRequest</p> <ul style="list-style-type: none">- triggerReason can only be <i>CablePluggedIn</i>- transactionInfo.chagringState should not be <i>Charging</i>- eventType must be <i>Updated</i> if TxStartPoint is <i>ParkingBayOccupancy</i>, else <i>Ended</i> |
| <p>Post scenario validations:</p> <p>N/a</p> |

TC_E_03_CS: Local start transaction - Cable plugin first - Success

| | |
|-------------------|---|
| Test case name | Local start transaction - Cable plugin first - Success |
| Test case Id | TC_E_03_CS |
| Use case Id(s) | E02 AND (C01 OR C02 OR C04 OR C06) |
| Requirement(s) | E02.FR.01, E02.FR.05, E02.FR.06, E02.FR.07, E02.FR.13, E02.FR.15, E02.FR.16, E02.FR.17, E01.FR.16 AND C01.FR.02, C02.FR.01, C06.FR.02 |
| System under test | Charging Station |
| Description | OCPP 2.x.x allows an EV driver to either first connect the EV and EVSE OR present a form of identification. Both sequences will result in being able to charge. |
| Purpose | To verify if the Charging Station is able to start a charging session when the EV driver first connects the EV and EVSE, before authorization. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The Charging Station supports at least one authorization method described at the following Use cases; C01, C02, C04, C06. - The Charging Station does NOT have a cable lock, which prevents the EV driver to connect the EV and EVSE before authorization. |

Before (Preparations)

Configuration State:
AuthCtrlr.AuthEnabled is *true* (If implemented AND ReadWrite)
AuthCtrlr.DisableRemoteAuthorization is *false* (If implemented)

Memory State:
N/a

Reusable State(s):
State is *EVConnectedPreSession*

Main (Test scenario)

| | |
|------------------|------|
| Charging Station | CSMS |
|------------------|------|

1. Execute **Reusable State** *Authorized* (local)

2. Execute **Reusable State** *EnergyTransferStarted*

Tool validations

N/a

Post scenario validations:
N/a

TC_E_04_CS: Local start transaction - Authorization first - Success

| | |
|-------------------|--|
| Test case name | Local start transaction - Authorization first - Success |
| Test case Id | TC_E_04_CS |
| Use case Id(s) | E03 AND (C01 OR C02 OR C04 OR C06) |
| Requirement(s) | E03.FR.01, E03.FR.06, E03.FR.12, E01.FR.16 AND C01.FR.02, C02.FR.01, C06.FR.02 |
| System under test | Charging Station |
| Description | OCPP 2.x.x allows an EV driver to either first connect the EV and EVSE OR present a form of identification. Both sequences will result in being able to charge. |
| Purpose | To verify if the Charging Station is able to start a charging session when the EV driver first presents a form of identification, before connecting the EV and EVSE. |
| Prerequisite(s) | The Charging Station supports at least one authorization method described at the following Use cases; C01, C02, C04, C06. |

| |
|---|
| Before (Preparations) |
| Configuration State: AuthCtrlr.AuthEnabled is <i>true</i> (If implemented AND ReadWrite) AuthCtrlr.DisableRemoteAuthorization is <i>false</i> (If implemented) |
| Memory State: N/a |
| Reusable State(s): State is <i>ParkingBayOccupied</i> (Optional state) |

| | |
|--|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>Authorized</i> (local) | |
| 2. Execute Reusable State <i>EnergyTransferStarted</i> | |

| |
|--|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_E_05_CS: Local start transaction - Authorization first - Cable plugin timeout

| | |
|-------------------|---|
| Test case name | Local start transaction - Authorization first - Cable plugin timeout |
| Test case Id | TC_E_05_CS |
| Use case Id(s) | E03 AND (C01 OR C02 OR C04 OR C06) |
| Requirement(s) | E03.FR.01, E03.FR.05, E03.FR.06, E03.FR.12 AND C01.FR.02, C02.FR.01, C06.FR.02 |
| System under test | Charging Station |
| Description | OCPP 2.x.x allows an EV driver to either first connect the EV and EVSE OR present a form of identification. Both sequences will result in being able to charge. |
| Purpose | To verify if the Charging Station is able to deauthorize the transaction after the EVConnectionTimeout has expired. |
| Prerequisite(s) | The Charging Station supports at least one authorization method described at the following Use cases; C01, C02, C04, C06. |

Before (Preparations)

Configuration State:

- TxCtrlr.EVConnectionTimeOut is <Configured ev_connection_timeout>
- AuthCtrlr.AuthEnabled is true (If implemented AND ReadWrite)
- AuthCtrlr.DisableRemoteAuthorization is false (If implemented)
- AuthCacheCtrlr.Enabled is false (If implemented)
- AuthCtrlr.LocalPreAuthorize is false

Memory State:

N/a

Reusable State(s):

State is *Authorized* (local)

Main (Test scenario)

| Charging Station | CSMS |
|--|--|
| 1. The Charging Station sends a TransactionEventRequest <u>Note(s):</u> - This step needs to be executed after the <Configured ev_connection_timeout> expires, if the transaction has been started. So in the case TxStartPoint contains <i>ParkingBayOccupancy</i> OR <i>Authorized</i> | 2. The Test System responds with a TransactionEventResponse |
| <u>Note(s):</u> - This step is only executed if TxStartPoint is <i>ParkingBayOccupancy</i> or <i>Authorized</i> - Optionally the Charging Station can send a <i>StatusNotificationRequest</i> or <i>NotifyEventRequest</i> with status <i>Available</i> | |
| 3. Execute Reusable State <i>Authorized</i> (local) <u>Note(s):</u> - This step is executed to verify if the EVSE is actually ready to start another charging session. | |
| 4. Execute Reusable State <i>EnergyTransferStarted</i> | |

Tool validations

* Step 1:

Message: **TransactionEventRequest**

- **triggerReason** must be *EVConnectTimeout*

If <Configured TxStopPoint> contains *Authorized* then**eventType** must be *Ended* AND**transactionInfo.stoppedReason** must be *Timeout*Else **eventType** must be *Updated*

Post scenario validations:

N/a

TC_E_38_CS: Local start transaction - EV not ready

| | |
|-------------------|---|
| Test case name | Local start transaction - EV not ready |
| Test case Id | TC_E_38_CS |
| Use case Id(s) | E03 |
| Requirement(s) | N/a |
| System under test | Charging Station |
| Description | OCPP 2.x.x allows an EV driver to either first connect the EV and EVSE OR present a form of identification. Both sequences will result in being able to charge. |
| Purpose | To verify if the Charging Station is able to handle and report if an EV is not ready to start the energy transfer (yet). |
| Prerequisite(s) | TxStartPoint should not be EnergyTransfer |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>Authorized</i> |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action</u> : Set the EV to a state in which it is NOT ready for energy transfer. | |
| 1. Execute Reusable State EVConnectedPreSession | |
| 2. The Charging Station sends a TransactionEventRequest | 3. The Test System responds with a TransactionEventResponse |

| |
|---|
| Tool validations |
| * Step 2: Message: TransactionEventRequest - triggerReason must be <i>ChargingStateChanged</i> - transactionInfo.chargingState should be <i>SuspendedEV</i> |
| Post scenario validations: N/a |

TC_E_52_CS: Local start transaction - Authorization first - DisableRemoteAuthorization

| | |
|-------------------|--|
| Test case name | Local start transaction - Authorization first - DisableRemoteAuthorization |
| Test case Id | TC_E_52_CS |
| Use case Id(s) | E03 AND C01 |
| Requirement(s) | C01.FR.02, C01.FR.05, |
| System under test | Charging Station |
| Description | When DisableRemoteAuthorization is set to true, the Charging Station will only try to look up an IdToken in Authorization Cache or Local Authorization List, and not do an AuthorizeRequest for IdTokens. This overrules requirement C01.FR.02 and C01.FR.05. |
| Purpose | To verify that the Charging Station will not send an AuthorizeRequest when DisableRemoteAuthorization is set to true. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The Charging Station supports Authorization Cache or the Local Authorization List - The Charging Station supports one of the following local authorization methods; RFID ISO14443 / RFID ISO15693 / KeyCode / MacAddress - AuthCtrl.DisableRemoteAuthorization is implemented. |

| |
|---|
| Before (Preparations) |
| Configuration State: AuthCtrl.Enabled is <i>true</i> (If implemented) AuthCtrl.DisableRemoteAuthorization is <i>true</i> |
| Memory State: None of the configured valid IdTokens is present in Authorization Cache or Local Authorization List. |
| Reusable State(s): State is <i>ParkingBayOccupied</i> (Optional state) |

| | |
|---|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action</u> : Present an idToken which is not configured in the Local Authorization List nor present in Authorization Cache. | |
| 1. The Charging Station does NOT send a AuthorizeRequest | |
| 2. Execute Reusable State <i>EVConnectedPreSession</i> | |
| 3. The Charging Station does NOT start charging. | |

| |
|--|
| Tool validations |
| * Step 1: Check that Charging Station does NOT send an AuthorizeRequest and authorization is refused. |
| Post scenario validations: - N/a |

TC_E_06_CS: Local Stop Transaction - Accepted

| | |
|-------------------|--|
| Test case name | Local Stop Transaction - Accepted |
| Test case Id | TC_E_06_CS |
| Use case Id(s) | E07 AND (C01 OR C02 OR C04) |
| Requirement(s) | E07.FR.04, E06.FR.15 AND C01.FR.03 |
| System under test | Charging Station |
| Description | The EV Driver is able to stop an ongoing transaction, by locally presenting an IdToken. |
| Purpose | To verify whether the Charging Station is able to perform a local stop authorization. |
| Prerequisite(s) | The Charging Station supports at least one authorization method described at the following Use cases; C01, C02, C04. |

| |
|---|
| Before (Preparations) |
| Configuration State: AuthCtrlr.AuthEnabled is <i>true</i> (If implemented AND ReadWrite) AuthCtrlr.DisableRemoteAuthorization is <i>false</i> (If implemented) |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> |

| | |
|---|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>StopAuthorized</i> (local) | |
| 2. Execute Reusable State <i>EVConnectedPostSession</i> | |
| 3. Execute Reusable State <i>EVDisconnected</i> | |
| 4. Execute Reusable State <i>ParkingBayUnoccupied</i> | |

| |
|--|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_E_07_CS: Stop transaction options - PowerPathClosed - Local stop

| | |
|-------------------|---|
| Test case name | Stop transaction options - PowerPathClosed - Local stop |
| Test case Id | TC_E_07_CS |
| Use case Id(s) | E06(S5) |
| Requirement(s) | E06.FR.06, E06.FR.15 |
| System under test | Charging Station |
| Description | OCPP 2.x.x has a flexible transaction mechanism that allows the transaction start and stop points to be configured differently. This test covers one of the stop options. |
| Purpose | To verify if the Charging Station stops a transaction when it is locally stopped by an EV driver and TxStopPoint contains <i>PowerPathClosed</i> . |
| Prerequisite(s) | <ul style="list-style-type: none"> - The Charging Station does NOT have the following configuration; The mutability of TxStopPoint is <i>ReadOnly</i> AND (the value <i>PowerPathClosed</i> is NOT set OR <i>Authorized</i> is set). - If the mutability of TxStopPoint is <i>ReadWrite</i>, then the value <i>PowerPathClosed</i> must be supported. |

| |
|---|
| Before (Preparations) |
| Configuration State: TxStopPoint contains <i>PowerPathClosed</i> |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> |

| | |
|---|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action</u> : Present <i>IdToken</i> to stop charging session. | |
| 1. Execute Reusable State <i>StopAuthorized</i> | |

| |
|--|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_E_35_CS: Stop transaction options - PowerPathClosed - Remote stop

| | |
|-------------------|---|
| Test case name | Stop transaction options - PowerPathClosed - Remote stop |
| Test case Id | TC_E_35_CS |
| Use case Id(s) | E06(S5) |
| Requirement(s) | E06.FR.06, E06.FR.15 |
| System under test | Charging Station |
| Description | OCPP 2.x.x has a flexible transaction mechanism that allows the transaction start and stop points to be configured differently. This test covers one of the stop options. |
| Purpose | To verify if the Charging Station stops a transaction when it is remotely stopped the CSMS and TxStopPoint contains <i>PowerPathClosed</i> . |
| Prerequisite(s) | <ul style="list-style-type: none"> - The Charging Station does NOT have the following configuration; The mutability of TxStopPoint is <i>ReadOnly</i> AND (the value <i>PowerPathClosed</i> is NOT set OR <i>Authorized</i> is set). - If the mutability of TxStopPoint is <i>ReadWrite</i>, then the value <i>PowerPathClosed</i> must be supported. |

Before (Preparations)

Configuration State:

TxStopPoint contains *PowerPathClosed*

Memory State:

N/a

Reusable State(s):

State is *EnergyTransferStarted*

Main (Test scenario)

| Charging Station | CSMS |
|--|---|
| 2. The Charging Station responds with a RequestStopTransactionResponse | 1. The Test System sends a RequestStopTransactionRequest with transactionId <transactionId provided by the Charging Station in <i>TransactionEventRequest</i> > |
| 3. The Charging Station sends a TransactionEventRequest | 4. The Test System responds with a TransactionEventResponse |
| <u>Note(s)</u> : The charging Station can stop a transaction in more than one <i>TransactionEventRequest</i> . | |

Tool validations

* Step 2:

Message: **RequestStopTransactionResponse**

- **status** must be *Accepted*

* Step 3:

Message: **TransactionEventRequest**

- **triggerReason** must be *RemoteStop* (for one of the *TransactionEventRequests*)

- **transactionInfo.stoppedReason** must be *Remote* (for the last *TransactionEventRequest*)

- **eventType** must be *Ended* (for the last *TransactionEventRequest*)

Post scenario validations:

N/a

TC_E_37_CS: Stop transaction options - PowerPathClosed - EV side disconnect

| | |
|-------------------|--|
| Test case name | Stop transaction options - PowerPathClosed - EV side disconnect |
| Test case Id | TC_E_37_CS |
| Use case Id(s) | E06(S5) |
| Requirement(s) | E06.FR.06, E06.FR.15 |
| System under test | Charging Station |
| Description | OCPP 2.x.x has a flexible transaction mechanism that allows the transaction start and stop points to be configured differently. This test covers one of the stop options. |
| Purpose | To verify if the Charging Station stops a transaction when the EV and the EVSE get disconnected and TxStopPoint contains <i>PowerPathClosed</i> . |
| Prerequisite(s) | <ul style="list-style-type: none"> - The Charging Station does NOT have the following configuration; The mutability of TxStopPoint is <i>ReadOnly</i> AND (the value <i>PowerPathClosed</i> is NOT set OR (<i>EnergyTransfer</i> OR <i>EVConnected</i> OR <i>DataSigned</i> is set)). - If the mutability of TxStopPoint is <i>ReadWrite</i>, then the value <i>PowerPathClosed</i> must be supported. |

Before (Preparations)

Configuration State:

TxStopPoint contains *PowerPathClosed*

StopTxOnEVSideDisconnect is *false* (If mutability is *ReadWrite*)

Memory State:

N/a

Reusable State(s):

State is *EnergyTransferSuspended*

Main (Test scenario)

| | |
|---|--|
| Charging Station | CSMS |
| <u>Manual Action</u> : Disconnect the EV and EVSE on EV side (EVSE loses connection with EV). | |
| 1. The Charging Station sends a TransactionEventRequest | 2. The Test System responds with a TransactionEventResponse |

Tool validations

* Step 1:

Message: **TransactionEventRequest**

- **triggerReason** must be *EVCommunicationLost*

- **transactionInfo.chargingState** must be *Idle*

- **transactionInfo.stoppedReason** must be *EVDIsconnected* or *StoppedByEV* (preferred value)

- **eventType** must be *Ended*

Post scenario validations:

N/a

TC_E_08_CS: Stop transaction options - EnergyTransfer stopped - StopAuthorized

| | |
|-------------------|---|
| Test case name | Stop transaction options - EnergyTransfer stopped - StopAuthorized |
| Test case Id | TC_E_08_CS |
| Use case Id(s) | E06(S6) |
| Requirement(s) | E06.FR.07, E06.FR.15 |
| System under test | Charging Station |
| Description | OCPP 2.x.x has a flexible transaction mechanism that allows the transaction start and stop points to be configured differently. This test covers one of the stop options. |
| Purpose | To verify if the Charging Station stops a transaction when the energy transfer stopped normally and it has been configured to do so. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The Charging Station does NOT have the following configuration; The mutability of TxStopPoint is <i>ReadOnly</i> AND (the value <i>EnergyTransfer</i> is NOT set OR (<i>Authorized</i> OR <i>PowerPathClosed</i>) is set). - If the mutability of TxStopPoint is <i>ReadWrite</i>, then the value <i>EnergyTransfer</i> must be supported. |

| |
|--|
| Before (Preparations) |
| Configuration State: TxStopPoint contains <i>EnergyTransfer</i> |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> |

| | |
|-----------------------------------|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. State is <i>StopAuthorized</i> | |

| |
|--|
| Tool validations |
| * Step 1: N/a |
| Post scenario validations: N/a |

TC_E_22_CS: Stop transaction options - EnergyTransfer stopped - SuspendedEV

| | |
|-------------------|---|
| Test case name | Stop transaction options - EnergyTransfer stopped - SuspendedEV |
| Test case Id | TC_E_22_CS |
| Use case Id(s) | E06(S6) |
| Requirement(s) | E06.FR.07, E06.FR.15 |
| System under test | Charging Station |
| Description | OCPP 2.x.x has a flexible transaction mechanism that allows the transaction start and stop points to be configured differently. This test covers one of the stop options. |
| Purpose | To verify if the Charging Station stops a transaction when the energy transfer stopped by the EV and the Charging Station has been configured to do so. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The Charging Station does NOT have the following configuration; The mutability of TxStopPoint is <i>ReadOnly</i> AND the value <i>EnergyTransfer</i> is NOT set. - If the mutability of TxStopPoint is <i>ReadWrite</i>, then the value <i>EnergyTransfer</i> must be supported. |

| |
|--|
| Before (Preparations) |
| Configuration State: TxStopPoint contains <i>EnergyTransfer</i> |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action:</u> <i>The EV suspends the energy transfer.</i> | |
| 1. The Charging Station sends a TransactionEventRequest | 2. The Test System responds with a TransactionEventResponse |

| |
|--|
| Tool validations |
| * Step 1: Message: TransactionEventRequest <ul style="list-style-type: none"> - triggerReason must be <i>ChargingStateChanged</i> - transactionInfo.chargingState must be <i>EVConnected</i> AND - transactionInfo.stoppedReason must be <i>StoppedByEV</i> AND - eventType must be <i>Ended</i> |
| Post scenario validations: N/a |

TC_E_14_CS: Stop transaction options - EVDisconnected - Charging Station side

| Test case name | Stop transaction options - EVDisconnected - Charging Station side |
|-------------------|---|
| Test case Id | TC_E_14_CS |
| Use case Id(s) | E06(S2) |
| Requirement(s) | E06.FR.02, E06.FR.15 |
| System under test | Charging Station |
| Description | OCPP 2.x.x has a flexible transaction mechanism that allows the transaction start and stop points to be configured differently. This test covers one of the stop options. |
| Purpose | To verify if the Charging Station stops a transaction when the EV and EVSE are disconnected at the Charging Station side and it has been configured to do so. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The Charging Station does NOT have the following configuration; The mutability of TxStopPoint is <i>ReadOnly</i> AND (the value <i>EVConnected</i> is NOT set OR (<i>EnergyTransfer</i> OR <i>PowerPathClosed</i> OR <i>DataSigned</i> OR <i>Authorized</i> is set)). - If the mutability of TxStopPoint is <i>ReadWrite</i>, then the value <i>EVConnected</i> must be supported. |

| Before (Preparations) |
|---|
| Configuration State: TxStopPoint contains <i>EVConnected</i> |
| Memory State: N/a |
| Reusable State(s): State is <i>EVConnectedPostSession</i> |

| Main (Test scenario) | |
|---|--|
| Charging Station | CSMS |
| <u>Manual Action</u> : Disconnect the EV and EVSE. | |
| 1. The Charging Station notifies the CSMS about the status change of the connector. | 2. The Test System responds accordingly. |
| 3. The Charging Station sends a TransactionEventRequest | 4. The Test System responds with a TransactionEventResponse |

| Tool validations |
|---|
| <p>* Step 1:</p> <p>Message: StatusNotificationRequest</p> <ul style="list-style-type: none"> - connectorStatus must be <i>Available</i> <p>Message: NotifyEventRequest</p> <ul style="list-style-type: none"> - eventData[0].trigger must be <i>Delta</i> - eventData[0].actualValue must be <i>Available</i> - eventData[0].component.name must be <i>Connector</i> - eventData[0].variable.name must be <i>AvailabilityState</i> <p>* Step 3:</p> <p>Message: TransactionEventRequest</p> <ul style="list-style-type: none"> - triggerReason must be <i>EVCommunicationLost</i> - transactionInfo.chargingState must be <i>Idle</i> <p>- If the Test System is configured to stop transactions using a <i>RequestStopTransactionRequest</i> message then transactionInfo.stoppedReason must be <i>Remote</i> or <i>EVDisconnected</i>.</p> <p>Else transactionInfo.stoppedReason must be <i>Local</i>, <i>EVDisconnected</i> or be omitted.</p> <ul style="list-style-type: none"> - eventType must be <i>Ended</i> |
| Post scenario validations: N/a |

TC_E_20_CS: Stop transaction options - EVDisconnected - EV side (able to charge IEC 61851-1 EV)

| | |
|-------------------|--|
| Test case name | Stop transaction options - EVDisconnected - EV side (able to charge IEC 61851-1 EV) |
| Test case Id | TC_E_20_CS |
| Use case Id(s) | E06(S2), E10 |
| Requirement(s) | E06.FR.02, E06.FR.15 |
| System under test | Charging Station |
| Description | OCPP 2.x.x has a flexible transaction mechanism that allows the transaction start and stop points to be configured differently. This test covers one of the stop options. |
| Purpose | To verify if the Charging Station stops a transaction when the EV and EVSE are disconnected at the EV side and it has been configured to do so. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The Charging Station does NOT have the following configuration; The mutability of TxStopPoint is <i>ReadOnly</i> AND (the value <i>EVConnected</i> is NOT set OR (<i>EnergyTransfer</i> OR <i>PowerPathClosed</i> is set)). - If the mutability of TxStopPoint is <i>ReadWrite</i>, then the value <i>EVConnected</i> must be supported. - The Charging Station does NOT have following configuration combination; StopTxOnEVSideDisconnect mutability <i>ReadOnly</i> with value <i>true</i> AND TxStopPoint mutability is <i>ReadOnly</i> and contains <i>Authorized</i>. - The Charging Station is able to charge with a EV that uses IEC 61851-1. |

| |
|---|
| Before (Preparations) |
| Configuration State: TxStopPoint contains <i>EVConnected</i> StopTxOnEVSideDisconnect is <i>false</i> (If mutability is <i>ReadWrite</i>) |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferSuspended</i> |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action</u> : Disconnect the EV and EVSE on EV side (EVSE loses connection with EV). | |
| 1. The Charging Station sends a TransactionEventRequest | 2. The Test System responds with a TransactionEventResponse |

| |
|--|
| Tool validations |
| * Step 1: Message: TransactionEventRequest - triggerReason must be <i>EVCommunicationLost</i> - transactionInfo.chargingState must be <i>Idle</i> - transactionInfo.stoppedReason must be <i>EVDisconnected</i> - eventType must be <i>Ended</i> |
| Post scenario validations: N/a |

TC_E_54_CS: Stop transaction options - EVDisconnected - EV side (not able to charge IEC 61851-1 EV)

| | |
|-------------------|--|
| Test case name | Stop transaction options - EVDisconnected - EV side (not able to charge IEC 61851-1 EV) |
| Test case Id | TC_E_54_CS |
| Use case Id(s) | E06(S2), E10 |
| Requirement(s) | E06.FR.02, E06.FR.15 |
| System under test | Charging Station |
| Description | OCPP 2.x.x has a flexible transaction mechanism that allows the transaction start and stop points to be configured differently. This test covers one of the stop options. |
| Purpose | To verify if the Charging Station stops a transaction when the EV and EVSE are disconnected at the EV side and it has been configured to do so. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The Charging Station does NOT have the following configuration; The mutability of TxStopPoint is <i>ReadOnly</i> AND (the value <i>EVConnected</i> is NOT set OR (<i>EnergyTransfer</i> OR <i>PowerPathClosed</i> is set)). - If the mutability of TxStopPoint is <i>ReadWrite</i>, then the value <i>EVConnected</i> must be supported. - The Charging Station does NOT have following configuration combination; StopTxOnEVSideDisconnect mutability <i>ReadOnly</i> with value <i>true</i> AND TxStopPoint mutability is <i>ReadOnly</i> and contains <i>Authorized</i>. - The Charging Station supports high level communication. |

| |
|---|
| Before (Preparations) |
| Configuration State: TxStopPoint contains <i>EVConnected</i> StopTxOnEVSideDisconnect is <i>false</i> (If mutability is <i>ReadWrite</i>) |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferSuspended</i> |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action:</u> <i>Disconnect the EV and EVSE on EV side (EVSE loses connection with EV).</i> | |
| 1. The Charging Station sends a TransactionEventRequest | 2. The Test System responds with a TransactionEventResponse |

| |
|--|
| Tool validations |
| * Step 1: Message: TransactionEventRequest - triggerReason must be <i>EVCommunicationLost</i> - transactionInfo.chargingState must be <i>Idle</i> - transactionInfo.stoppedReason must be <i>StoppedByEV</i> or <i>EVDisconnected</i> - eventType must be <i>Ended</i> |
| Post scenario validations: N/a |

TC_E_15_CS: Stop transaction options - StopAuthorized - Local

| | |
|-------------------|--|
| Test case name | Stop transaction options - StopAuthorized - Local |
| Test case Id | TC_E_15_CS |
| Use case Id(s) | E06(S3) |
| Requirement(s) | E06.FR.03, E06.FR.15 |
| System under test | Charging Station |
| Description | OCPP 2.x.x has a flexible transaction mechanism that allows the transaction start and stop points to be configured differently. This test covers one of the stop options. |
| Purpose | To verify if the Charging Station stops a transaction when the EV driver locally stops the transaction and it has been configured to do so. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The Charging Station supports at least one authorization method described at the following Use cases; C01, C02, C04. - The Charging Station does NOT have the following configuration; The mutability of TxStopPoint is <i>ReadOnly</i> AND the value <i>Authorized</i> is NOT set OR <i>PowerPathClosed</i> is set. - If the mutability of TxStopPoint is <i>ReadWrite</i>, then the value <i>Authorized</i> must be supported. |

| |
|--|
| Before (Preparations) |
| Configuration State: TxStopPoint contains <i>Authorized</i> |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Notes(s)</u> : The tool will wait for <Configured Transaction Duration> seconds | |
| <u>Manual Action</u> : Present IdToken to stop charging session. | |
| 1. The Charging Station sends a TransactionEventRequest | 2. The Test System responds with a TransactionEventResponse |

| |
|--|
| Tool validations |
| * Step 1: Message: TransactionEventRequest - triggerReason must be <i>StopAuthorized</i> - transactionInfo.stoppedReason must be <i>Local</i> - eventType must be <i>Ended</i> |
| Post scenario validations: N/a |

TC_E_21_CS: Stop transaction options - StopAuthorized - Remote

| | |
|-------------------|--|
| Test case name | Stop transaction options - StopAuthorized - Remote |
| Test case Id | TC_E_21_CS |
| Use case Id(s) | E06(S3) AND F03 |
| Requirement(s) | E06.FR.03, E06.FR.15 AND F03.FR.09 |
| System under test | Charging Station |
| Description | OCPP 2.x.x has a flexible transaction mechanism that allows the transaction start and stop points to be configured differently. This test covers one of the stop options. |
| Purpose | To verify if the Charging Station stops a transaction when it receives a RequestStopTransactionRequest and it has been configured to do so. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The Charging Station does NOT have the following configuration; The mutability of TxStopPoint is <i>ReadOnly</i> AND the value <i>Authorized</i> is NOT set OR <i>PowerPathClosed</i> is set. - If the mutability of TxStopPoint is <i>ReadWrite</i>, then the value <i>Authorized</i> must be supported. |

Before (Preparations)

Configuration State:
TxStopPoint contains *Authorized*

Memory State:
 N/a

Reusable State(s):
 State is *EnergyTransferStarted*

Main (Test scenario)

| Charging Station | CSMS |
|---|---|
| 2. The Charging Station responds with a RequestStopTransactionResponse | 1. The Test System sends a RequestStopTransactionRequest with transactionId <transactionId provided by the Charging Station in <i>TransactionEventRequest</i> > |
| 3. The Charging Station sends a TransactionEventRequest | 4. The Test System responds with a TransactionEventResponse |

Tool validations

* Step 2:

Message: **RequestStopTransactionResponse**

- **status** must be *Accepted*

* Step 3:

Message: **TransactionEventRequest**

- **triggerReason** must be *RemoteStop*

- **transactionInfo.stoppedReason** must be *Remote*

- **eventType** must be *Ended*

Post scenario validations:

N/a

TC_E_19_CS: Stop transaction options - ParkingBayUnoccupied

| | |
|-------------------|---|
| Test case name | Stop transaction options - ParkingBayUnoccupied |
| Test case Id | TC_E_19_CS |
| Use case Id(s) | E06(S1) |
| Requirement(s) | E06.FR.01, E06.FR.15 |
| System under test | Charging Station |
| Description | OCPP 2.x.x has a flexible transaction mechanism that allows the transaction start and stop points to be configured differently. This test covers one of the stop options. |
| Purpose | To verify if the Charging Station stops a transaction when the EV left the parking bay and it has been configured to do so. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The Charging Station does NOT have the following configuration; The mutability of TxStopPoint is <i>ReadOnly</i> AND (the value <i>ParkingBayOccupancy</i> is NOT set OR (<i>EnergyTransfer</i> OR <i>PowerPathClosed</i> OR <i>DataSigned</i> OR <i>Authorized</i> OR <i>EVConnected</i> is set)). - If the mutability of TxStopPoint is <i>ReadWrite</i>, then the value <i>ParkingBayOccupancy</i> must be supported. |

| |
|---|
| Before (Preparations) |
| Configuration State: TxStopPoint contains <i>ParkingBayOccupancy</i> |
| Memory State: N/a |
| Reusable State(s): State is <i>EVDDisconnected</i> |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action</u> : Drive EV out of parking bay. | |
| 1. The Charging Station sends a TransactionEventRequest | 2. The Test System responds with a TransactionEventResponse |

| |
|---|
| Tool validations |
| * Step 1: Message: TransactionEventRequest - triggerReason must be <i>EVDeparted</i> - If the Test System is configured to stop transactions using a <i>RequestStopTransactionRequest</i> message then transactionInfo.stoppedReason must be <i>Remote</i> Else transactionInfo.stoppedReason must be <i>Local</i> - eventType must be <i>Ended</i> |
| Post scenario validations: N/a |

TC_E_24_CS: Disconnect cable on EV-side - Deauthorize transaction - UnlockOnEVSideDisconnect is true

| | |
|-------------------|---|
| Test case name | Disconnect cable on EV-side - Deauthorize transaction - UnlockOnEVSideDisconnect is true |
| Test case Id | TC_E_24_CS |
| Use case Id(s) | E09 |
| Requirement(s) | E09.FR.01, E09.FR.02, E09.FR.04 |
| System under test | Charging Station |
| Description | The Charging Station can behave in several different ways when the cable is disconnected at the EV side, based on its configuration. This test case tests one of the possible configuration settings. |
| Purpose | To verify if the Charging Station deauthorizes the transaction when the EV and EVSE are disconnected at the EV side and it has been configured to do so AND unlocks the cable at Charging Station side. |
| Prerequisite(s) | The Charging Station does NOT have a permanently attached cable. |

| |
|---|
| Before (Preparations) |
| Configuration State: StopTxOnEVSideDisconnect is <i>true</i> UnlockOnEVSideDisconnect is <i>true</i> |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferSuspended</i> |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action:</u> Disconnect the EV and EVSE on EV side (EVSE loses connection with EV). | |
| 1. The Charging Station sends a TransactionEventRequest | 2. The Test System responds with a TransactionEventResponse |
| <u>Manual Action:</u> Disconnect the EV and EVSE on Charging Station side. | |
| 3. The Charging Station notifies the CSMS about the current state of the connector. | 4. The Test System responds accordingly. |

| |
|---|
| Tool validations |
| <p>* Step 1:</p> <p>Message: TransactionEventRequest</p> <ul style="list-style-type: none"> - triggerReason must be <i>EVCommunicationLost</i> - transactionInfo.chargingState must be <i>Idle</i> <p>* Step 3:</p> <p>Message: StatusNotificationRequest</p> <ul style="list-style-type: none"> - connectorStatus <i>Available</i> <p>Message: NotifyEventRequest</p> <ul style="list-style-type: none"> - eventData[0].trigger <i>Delta</i> - eventData[0].actualValue <i>"Available"</i> - eventData[0].component.name <i>"Connector"</i> - eventData[0].variable.name <i>"AvailabilityState"</i> <p>Post scenario validations:</p> <p>N/a</p> |

TC_E_25_CS: Disconnect cable on EV-side - Deauthorize transaction - UnlockOnEVSideDisconnect is false

| | |
|-------------------|--|
| Test case name | Disconnect cable on EV-side - Deauthorize transaction - UnlockOnEVSideDisconnect is false |
| Test case Id | TC_E_25_CS |
| Use case Id(s) | E09 |
| Requirement(s) | E09.FR.01, E09.FR.03, E09.FR.04 |
| System under test | Charging Station |
| Description | The Charging Station can behave in several different ways when the cable is disconnected at the EV side, based on its configuration. This test case tests one of the possible configuration settings. |
| Purpose | To verify if the Charging Station deauthorizes the transaction when the EV and EVSE are disconnected at the EV side and it has been configured to do so AND keeps the cable locked at Charging Station side. |
| Prerequisite(s) | N/a |

| |
|--|
| Before (Preparations) |
| Configuration State: StopTxOnEVSideDisconnect is <i>true</i> UnlockOnEVSideDisconnect is <i>false</i> |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferSuspended</i> |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action:</u> Disconnect the EV and EVSE on EV side (EVSE loses connection with EV). | |
| 1. The Charging Station sends a TransactionEventRequest | 2. The Test System responds with a TransactionEventResponse |
| <u>Manual Action:</u> Present the IdToken that was used to start the transaction. | |
| <u>Note(s):</u> - This manual action needs to be executed when the Charging Station has a detachable cable on the Charging Station side. | |
| <u>Manual Action:</u> Disconnect the EV and EVSE on Charging Station side. | |
| <u>Note(s):</u> - This manual action needs to be executed when the Charging Station has a detachable cable on the Charging Station side. | |
| 3. The Charging Station notifies the CSMS about the current state of the connector. | 4. The Test System responds accordingly. |

| |
|---|
| Tool validations |
| * Step 1: Message: TransactionEventRequest - triggerReason must be <i>EVCommunicationLost</i> - transactionInfo.chargingState must be <i>Idle</i> * Step 3: Message: StatusNotificationRequest - connectorStatus <i>Available</i> Message: NotifyEventRequest - eventData[0].trigger <i>Delta</i> - eventData[0].actualValue <i>"Available"</i> - eventData[0].component.name <i>"Connector"</i> - eventData[0].variable.name <i>"AvailabilityState"</i> |

| |
|--|
| Tool validations |
| Post scenario validations: N/a |

TC_E_26_CS: Disconnect cable on EV-side - Suspend transaction

| | |
|--------------------------|--|
| Test case name | Disconnect cable on EV-side - Suspend transaction |
| Test case Id | TC_E_26_CS |
| Use case Id(s) | E10 |
| Requirement(s) | E10.FR.01, E10.FR.03 |
| System under test | Charging Station |
| Description | The Charging Station can behave in several different ways when the cable is disconnected at the EV side, based on its configuration. This test case tests one of the possible configuration settings. |
| Purpose | To verify if the Charging Station suspends the transaction when the EV and EVSE are disconnected at the EV side and it has been configured to do so AND is able restart the energy transfer after reconnecting the EV and EVSE. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The Charging Station does NOT have the following configuration; The mutability of TxStopPoint is <i>ReadOnly</i> AND (the value <i>Authorized</i> OR <i>ParkingBayOccupancy</i> is NOT set OR (<i>EnergyTransfer</i> OR <i>PowerPathClosed</i> OR <i>DataSigned</i> OR <i>EVConnected</i> is set)). - If the mutability of TxStopPoint is <i>ReadWrite</i>, then the value <i>Authorized</i> OR <i>ParkingBayOccupancy</i> must be supported. |

Before (Preparations)

Configuration State:

TxStopPoint contains *Authorized* (If supported) AND/OR *ParkingBayOccupancy* (If supported)

UnlockOnEVSideDisconnect is *false*

StopTxOnEVSideDisconnect is *false*

Memory State:

N/a

Reusable State(s):

State is *EnergyTransferSuspended*

Main (Test scenario)

| Charging Station | CSMS |
|---|--|
| <u>Manual Action:</u> Disconnect the EV and EVSE on EV side (EVSE loses connection with EV). | |
| 1. The Charging Station sends a TransactionEventRequest | 2. The Test System responds with a TransactionEventResponse |
| 3. The Charging Station notifies the CSMS about the current state of the connector. | 4. The Test System responds accordingly. |
| <u>Note(s):</u> - This step needs to be executed when the Charging Station has a permanently attached cable on the Charging Station side. | |
| <u>Manual Action:</u> Reconnect the EV and EVSE on EV side. | |
| <u>Note(s):</u> - If the Charging Station has a permanently attached cable on the Charging Station side, then this step needs to be executed before the configured <i>EVConnectionTimeout</i> expires. | |
| 5. The Charging Station sends a TransactionEventRequest | 6. The Test System responds with a TransactionEventResponse |
| 7. The Charging Station sends a TransactionEventRequest | 8. The Test System responds with a TransactionEventResponse |

| Tool validations |
|--|
| <p>* Step 1:</p> <p>Message: TransactionEventRequest</p> <ul style="list-style-type: none">- triggerReason must be <i>EVCommunicationLost</i>- transactionInfo.chargingState must be <i>Idle</i>- eventType must be <i>Updated</i> <p>* Step 3:</p> <p>Message: StatusNotificationRequest</p> <ul style="list-style-type: none">- connectorStatus <i>Available</i> <p>Message: NotifyEventRequest</p> <ul style="list-style-type: none">- eventData[0].trigger <i>Delta</i>- eventData[0].actualValue <i>"Available"</i>- eventData[0].component.name <i>"Connector"</i>- eventData[0].variable.name <i>"AvailabilityState"</i> <p>* Step 5:</p> <p>Message: TransactionEventRequest</p> <ul style="list-style-type: none">- triggerReason must be <i>CablePluggedIn</i>- transactionInfo.chargingState must be <i>EVConnected</i>- eventType must be <i>Updated</i> <p>* Step 7:</p> <p>Message: TransactionEventRequest</p> <ul style="list-style-type: none">- triggerReason must be <i>ChargingStateChanged</i>- transactionInfo.chargingState must be <i>Charging</i>- eventType must be <i>Updated</i> |
| <p>Post scenario validations:</p> <p>N/a</p> |

TC_E_27_CS: Disconnect cable on EV-side - Suspend transaction - Fixed cable connection timeout

| | |
|-------------------|---|
| Test case name | Disconnect cable on EV-side - Suspend transaction - Fixed cable connection timeout |
| Test case Id | TC_E_27_CS |
| Use case Id(s) | E10 |
| Requirement(s) | E10.FR.02, E10.FR.03 |
| System under test | Charging Station |
| Description | The Charging Station can behave in several different ways when the cable is disconnected at the EV side, based on its configuration. This test case tests one of the possible configuration settings. |
| Purpose | To verify if the Charging Station suspends the transaction when the EV and EVSE are disconnected at the EV side and it has been configured to do so AND deauthorizes the transaction after the configured connection timeout expires. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The Charging Station does NOT have the following configuration; The mutability of TxStopPoint is <i>ReadOnly</i> AND (the value <i>Authorized</i> OR <i>ParkingBayOccupancy</i> is NOT set OR (<i>EnergyTransfer</i> OR <i>PowerPathClosed</i> OR <i>DataSigned</i> OR <i>EVConnected</i> is set)). - If the mutability of TxStopPoint is <i>_ReadWrite</i>, then the value <i>Authorized</i> OR <i>ParkingBayOccupancy</i> must be supported. - The Charging Station has a permanently attached cable at the Charging Station side. - StopTxOnEVSideDisconnect can be set to <i>false</i>. |

| |
|--|
| Before (Preparations) |
| Configuration State: TxStopPoint contains <i>Authorized</i> (If supported) TxStopPoint contains <i>ParkingBayOccupancy</i> (If supported) UnlockOnEVSideDisconnect is <i>false</i> StopTxOnEVSideDisconnect is <i>false</i> |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferSuspended</i> |

| Main (Test scenario) | |
|---|--|
| Charging Station | CSMS |
| <u>Manual Action:</u> <i>Disconnect the EV and EVSE on EV side (EVSE loses connection with EV).</i> | |
| 1. The Charging Station sends a TransactionEventRequest | 2. The Test System responds with a TransactionEventResponse |
| 3. The Charging Station notifies the CSMS about the current state of the connector. | 4. The Test System responds accordingly. |
| 5. The Charging Station sends a TransactionEventRequest | 6. The Test System responds with a TransactionEventResponse |

| Tool validations |
|---|
| <p>* Step 1:</p> <p>Message: TransactionEventRequest</p> <ul style="list-style-type: none">- triggerReason must be <i>EVCommunicationLost</i>- transactionInfo.chargingState must be <i>Idle</i>- eventType must be <i>Updated</i> <p>* Step 3:</p> <p>Message: StatusNotificationRequest</p> <ul style="list-style-type: none">- connectorStatus <i>Available</i> <p>Message: NotifyEventRequest</p> <ul style="list-style-type: none">- eventData[0].trigger <i>Delta</i>- eventData[0].actualValue <i>"Available"</i>- eventData[0].component.name <i>"Connector"</i>- eventData[0].variable.name <i>"AvailabilityState"</i> <p>* Step 5:</p> <p>Message: TransactionEventRequest</p> <ul style="list-style-type: none">- triggerReason must be <i>EVConnectTimeout</i> <p>If <Configured TxCtrlr.TxStopPoint> contains <i>Authorized</i> then</p> <p>eventType must be <i>Ended</i></p> <p>transactionInfo.stoppedReason must be <i>Timeout</i></p> <p>else if <Configured TxCtrlr.TxStopPoint> contains <i>ParkingBayOccupancy</i> then</p> <p>eventType must be <i>Updated</i></p> |
| <p>Post scenario validations:</p> <p>N/a</p> |

TC_E_28_CS: Check Transaction status - TransactionId unknown

| | |
|-------------------|--|
| Test case name | Check Transaction status - TransactionId unknown |
| Test case Id | TC_E_28_CS |
| Use case Id(s) | E14 |
| Requirement(s) | E14.FR.01 |
| System under test | Charging Station |
| Description | The CSMS is able to request the status of a transaction and to find out whether there are queued transaction-related messages, using the GetTransactionStatusRequest message. |
| Purpose | To verify if the Charging Station is able to handle receiving a GetTransactionStatusRequest for an unknown transactionId. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a GetTransactionStatusResponse | 1. The Test System sends a GetTransactionStatusRequest with transactionId <Randomly generated transactionId> |

| |
|--|
| Tool validations |
| * Step 2: Message: GetTransactionStatusResponse - ongoingIndicator must be <i>false</i> - messagesInQueue must be <i>false</i> |
| Post scenario validations: N/a |

TC_E_29_CS: Check Transaction status - Transaction with id ongoing - with message in queue

| | |
|-------------------|---|
| Test case name | Check Transaction status - Transaction with id ongoing - with message in queue |
| Test case Id | TC_E_29_CS |
| Use case Id(s) | E14 |
| Requirement(s) | E14.FR.02,E14.FR.04 |
| System under test | Charging Station |
| Description | The CSMS is able to request the status of a transaction and to find out whether there are queued transaction-related messages, using the GetTransactionStatusRequest message. |
| Purpose | To verify if the Charging Station is able to correctly respond to a GetTransactionStatusRequest with a transactionId, while there is a message queued belonging to an ongoing transaction with the requested id. |
| Prerequisite(s) | N/a |

| |
|---|
| Before (Preparations) |
| <p>Configuration State:</p> <p>SampledDataTxUpdatedMeasurands is <Configured transaction_updated_metervalues_measurands></p> <p>SampledDataTxUpdatedInterval is <Configured transaction_updated_metervalues_interval></p> <p>OfflineThreshold is <Configured RetryBackOffWaitMinimum_duration> + 60.0</p> <p>RetryBackOffWaitMinimum is <Configured RetryBackOffWaitMinimum_duration></p> <p>RetryBackOffRandomRange is 0</p> <p><u>Note:</u> <Configured RetryBackOffWaitMinimum_duration> must be greater than <Configured Transaction MeterValues interval></p> <p>Memory State: N/a</p> <p>Reusable State(s): State is <i>EnergyTransferStarted</i></p> |

| Main (Test scenario) | |
|---|--|
| Charging Station | CSMS |
| | 1. The Test System closes the WebSocket connection AND does not accept a reconnect. |
| | 2. The Test System waits a number of seconds equal to <Configured RetryBackOffWaitMinimum_duration>, before accepting a reconnection attempt from the Charging Station. |
| 4. The Charging Station responds with a GetTransactionStatusResponse | 3. The Test System sends a GetTransactionStatusRequest with transactionId <Generated transactionId from Before> <u>Note:</u> This step will be executed the moment the WebSocket connection is restored. |
| 5. The Charging Stations sends a TransactionEventRequest <u>Note(s):</u> - The Charging Station will empty its Transaction message queue. This will contain one or more TransactionEventRequest messages | 6. The Test System responds with a TransactionEventResponse |

| Tool validations |
|--|
| <div>* Step 4: Message: GetTransactionStatusResponse<ul style="list-style-type: none">- ongoingIndicator must be <i>true</i>- messagesInQueue must be <i>true</i></div> <div>* Step 5: Message: TransactionEventRequest<ul style="list-style-type: none">- eventType must be <i>Updated</i>- meterValues must be present.- offline must be <i>true</i></div> |
| <div>Post scenario validations: N/a</div> |

TC_E_30_CS: Check Transaction status - Transaction with id ongoing - without message in queue

| | |
|-------------------|--|
| Test case name | Check Transaction status - Transaction with id ongoing - without message in queue |
| Test case Id | TC_E_30_CS |
| Use case Id(s) | E14 |
| Requirement(s) | E14.FR.02,E14.FR.05 |
| System under test | Charging Station |
| Description | The CSMS is able to request the status of a transaction and to find out whether there are queued transaction-related messages, using the GetTransactionStatusRequest message. |
| Purpose | To verify if the Charging Station is able to correctly respond to a GetTransactionStatusRequest with a transactionId, while there is NO message queued belonging to an ongoing transaction with the requested id. |
| Prerequisite(s) | N/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a GetTransactionStatusResponse | 1. The Test System sends a GetTransactionStatusRequest with transactionId <Generated transactionId from Before> |

| |
|---|
| Tool validations |
| * Step 2: Message: GetTransactionStatusResponse - ongoingIndicator must be <i>true</i> - messagesInQueue must be <i>false</i> |
| Post scenario validations: N/a |

TC_E_31_CS: Check Transaction status - Transaction with id ended - with message in queue

| | |
|-------------------|---|
| Test case name | Check Transaction status - Transaction with id ended - with message in queue |
| Test case Id | TC_E_31_CS |
| Use case Id(s) | E14 |
| Requirement(s) | E14.FR.03,E14.FR.04 |
| System under test | Charging Station |
| Description | The CSMS is able to request the status of a transaction and to find out whether there are queued transaction-related messages, using the GetTransactionStatusRequest message. |
| Purpose | To verify if the Charging Station is able to correctly respond to a GetTransactionStatusRequest with a transactionId, while there is a message queued belonging to an ended transaction with the requested id. |
| Prerequisite(s) | <p>The following combination of conditions are NOT true:</p> <ul style="list-style-type: none"> - No local authorization methods are supported AND - TxStopPoint mutability is <i>false</i> and only contains Authorized AND - TxCtrlr.StopTxOnEVSideDisconnect mutability is <i>false</i> and value is <i>false</i> <p>Note: If conditions 2 and 3 are true, but condition 1 is false, then please configure Test System configuration <scenario> as local.</p> |

| |
|--|
| Before (Preparations) |
| <p>Configuration State:</p> <p>SampledDataTxUpdatedMeasurands is <Configured transaction_updated_metervalues_measurands></p> <p>SampledDataTxUpdatedInterval is <Configured transaction_updated_metervalues_interval></p> <p>OfflineThreshold is <Configured RetryBackOffWaitMinimum_duration> + <Configured Transaction Duration> + 60.0</p> <p>RetryBackOffWaitMinimum is <Configured RetryBackOffWaitMinimum_duration> + <Configured Transaction Duration></p> <p>RetryBackOffRandomRange is 0</p> <p>UnlockOnEVSideDisconnect is true (If ReadWrite)</p> <p><u>Note:</u> <Configured Transaction Duration> should be long enough to execute manual tasks</p> <p>Memory State: N/a</p> <p>Reusable State(s): State is <i>EnergyTransferStarted</i></p> |

| Main (Test scenario) | |
|---|---|
| Charging Station | CSMS |
| | <i>The Test System closes the WebSocket connection AND does not accept a reconnect.</i> |
| <u>Manual Action</u> : Present the same idToken as used to start the transaction. <u>Notes(s)</u> : Only if configured scenario is local | |
| <u>Manual Action</u> : Stop the energy transfer via the EV. <u>Notes(s)</u> : Only if configured scenario is remote | |
| <u>Manual Action</u> : Disconnect the EV and EVSE. | |
| <u>Manual Action</u> : Drive EV out of parking bay. (Only needed if TxStopPoint is ParkingBayOccupancy) | |
| <u>Notes(s)</u> : The tool will wait for <Configured Transaction Duration> seconds | |
| | <i>The Test System accepts reconnection attempt from the Charging Station.</i> |
| 2. The Charging Station responds with a GetTransactionStatusResponse | 1. The Test System sends a GetTransactionStatusRequest with transactionId <Generated transactionId from Before> <u>Note</u> : <i>This step will be executed the moment the WebSocket connection is restored.</i> |

| Main (Test scenario) | |
|--|---|
| 3. The Charging Stations sends a <code>TransactionEventRequest</code> <u>Note(s):</u> - The Charging Station will empty its <i>Transaction message queue</i> . This will contain all <i>TransactionEventRequest</i> messages from the <i>Transaction</i> . | 4. The Test System responds with a <code>TransactionEventResponse</code> |

| Tool validations |
|---|
| <p>* Step 2:</p> <p>Message: <code>GetTransactionStatusResponse</code></p> <ul style="list-style-type: none"> - <code>ongoingIndicator</code> must be <i>false</i> - <code>messagesInQueue</code> must be <i>true</i> <p>* Step 3:</p> <p>Message: <code>TransactionEventRequest</code></p> <p>The tool validations from the reusable states need to be used to verify whether all required <i>TransactionEventRequests</i> have been received.</p> <p>From <i>StopAuthorized</i> through <i>ParkingBayUnoccupied</i> (in case of scenario <i>Local</i>).</p> <p>And from <i>EnergyTransferSuspended</i> through <i>ParkingBayUnoccupied</i> (in case of scenario <i>Remote</i>).</p> |
| Post scenario validations: N/a |

TC_E_32_CS: Check Transaction status - Transaction with id ended - without message in queue

| | |
|-------------------|--|
| Test case name | Check Transaction status - Transaction with id ended - without message in queue |
| Test case Id | TC_E_32_CS |
| Use case Id(s) | E14 |
| Requirement(s) | E14.FR.03,E14.FR.05 |
| System under test | Charging Station |
| Description | The CSMS is able to request the status of a transaction and to find out whether there are queued transaction-related messages, using the GetTransactionStatusRequest message. |
| Purpose | To verify if the Charging Station is able to correctly respond to a GetTransactionStatusRequest with a transactionId, while there is NO message queued belonging to an ended transaction with the requested id. |
| Prerequisite(s) | N/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> , <i>ParkingBayUnoccupied</i> |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a GetTransactionStatusResponse | 1. The Test System sends a GetTransactionStatusRequest with transactionId <Generated transactionId from Before> |

| |
|--|
| Tool validations |
| * Step 2: Message: GetTransactionStatusResponse - ongoingIndicator must be <i>false</i> - messagesInQueue must be <i>false</i> |
| Post scenario validations: N/a |

TC_E_33_CS: Check Transaction status - Without transactionId - with message in queue

| | |
|-------------------|--|
| Test case name | Check Transaction status - Without transactionId - with message in queue |
| Test case Id | TC_E_33_CS |
| Use case Id(s) | E14 |
| Requirement(s) | E14.FR.06,E14.FR.07 |
| System under test | Charging Station |
| Description | The CSMS is able to request the status of a transaction and to find out whether there are queued transaction-related messages, using the GetTransactionStatusRequest message. |
| Purpose | To verify if the Charging Station is able to correctly respond to a GetTransactionStatusRequest without a transactionId, while there is a message queued. |
| Prerequisite(s) | N/a |

| |
|--|
| Before (Preparations) |
| <p>Configuration State: SampledDataTxUpdatedMeasurands is <Configured transaction_updated_metervalues_measurands> SampledDataTxUpdatedInterval is <Configured transaction_updated_metervalues_interval> OfflineThreshold is <Configured RetryBackOffWaitMinimum_duration> + 60.0 RetryBackOffWaitMinimum is <Configured RetryBackOffWaitMinimum_duration> RetryBackOffRandomRange is 0 <u>Note:</u> <Configured RetryBackOffWaitMinimum_duration> must be greater than <Configured Transaction MeterValues interval></p> <p>Memory State: N/a</p> <p>Reusable State(s): State is <i>EnergyTransferStarted</i></p> |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| | 1. The Test System closes the WebSocket connection AND does not accept a reconnect. |
| | 2. The Test System waits a number of seconds equal to <Configured RetryBackOffWaitMinimum_duration>, before accepting a reconnection attempt from the Charging Station. |
| 4. The Charging Station responds with a GetTransactionStatusResponse | 3. The Test System sends a GetTransactionStatusRequest with transactionId omitted <u>Note:</u> This step will be executed the moment the WebSocket connection is restored. |
| 5. The Charging Stations sends a TransactionEventRequest | 6. The Test System responds with a TransactionEventResponse |
| <u>Note(s):</u> - The Charging Station will empty its Transaction message queue. This will contain one or more TransactionEventRequest messages | |

| Tool validations |
|--|
| <p>* Step 4:</p> <p>Message: GetTransactionStatusResponse</p> <ul style="list-style-type: none">- ongoingIndicator must be omitted- messagesInQueue must be <i>true</i> <p>* Step 5:</p> <p>Message: TransactionEventRequest</p> <ul style="list-style-type: none">- eventType must be <i>Updated</i>- meterValues must be present.- offline must be <i>true</i> |
| <p>Post scenario validations:</p> <p>N/a</p> |

TC_E_34_CS: Check Transaction status - Without transactionId - without message in queue

| | |
|-------------------|--|
| Test case name | Check Transaction status - Without transactionId - without message in queue |
| Test case Id | TC_E_34_CS |
| Use case Id(s) | E14 |
| Requirement(s) | E14.FR.06,E14.FR.08 |
| System under test | Charging Station |
| Description | The CSMS is able to request the status of a transaction and to find out whether there are queued transaction-related messages, using the GetTransactionStatusRequest message. |
| Purpose | To verify if the Charging Station is able to correctly respond to a GetTransactionStatusRequest without a transactionId, while there is NO message queued. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a GetTransactionStatusResponse | 1. The Test System sends a GetTransactionStatusRequest with transactionId omitted |

| |
|---|
| Tool validations |
| * Step 2: Message: GetTransactionStatusResponse - ongoingIndicator must be omitted - messagesInQueue must be <i>false</i> |
| Post scenario validations: N/a |

TC_E_40_CS: Offline Behaviour - Connection loss during transaction

| | |
|-------------------|--|
| Test case name | Offline Behaviour - Connection loss during transaction |
| Test case Id | TC_E_40_CS |
| Use case Id(s) | E11 |
| Requirement(s) | E11.FR.01,E11.FR.02,E11.FR.06 |
| System under test | Charging Station |
| Description | The Charging Station queues TransactionEvent messages to inform the CSMS that a transaction occurred while the Charging Station was Offline. |
| Purpose | To verify if the Charging Station is able to queue TransactionEvent messages while it is offline. |
| Prerequisite(s) | N/a |

Before (Preparations)

Configuration State:

SampledDataTxUpdatedMeasurands is <Configured transaction_updated_metervalues_measurands>

SampledDataTxUpdatedInterval is <Configured transaction_updated_metervalues_interval>

SampledDataEnabled is true

OfflineThreshold is <Configured RetryBackOffWaitMinimum_duration> + 60.0

RetryBackOffWaitMinimum is <Configured RetryBackOffWaitMinimum_duration>

RetryBackOffRandomRange is 0

Note:

<Configured RetryBackOffWaitMinimum_duration> must be greater than <Configured Transaction MeterValues interval>

Memory State:

N/a

Reusable State(s):

State is *EnergyTransferStarted*

Main (Test scenario)

| Charging Station | CSMS |
|---|---|
| | 1. The Test System closes the WebSocket connection AND does not accept a reconnect. |
| | 2. The Test System waits a number of seconds equal to <Configured RetryBackOffWaitMinimum_duration>, before accepting a reconnection attempt from the Charging Station. |
| 3. The Charging Stations sends a TransactionEventRequest | 4. The Test System responds with a TransactionEventResponse |
| Note(s): - The Charging Station will empty its Transaction message queue. This will contain one or more TransactionEventRequest messages | |

Tool validations

* Step 3:

All messages: **TransactionEventRequest**

- **eventType** must be *Updated*
- **meterValues** must be present.
- **offline** must be *true*

Post scenario validations:

N/a

TC_E_41_CS: Retry sending transaction message when failed - Max retry count reached

| | |
|--------------------------|--|
| Test case name | Retry sending transaction message when failed - Max retry count reached |
| Test case Id | TC_E_41_CS |
| Use case Id(s) | E13 |
| Requirement(s) | E13.FR.01,E13.FR.02,E13.FR.03,E13.FR.04 |
| System under test | Charging Station |
| Description | There are situations/issues why a CSMS might not accept a transaction related message, or does not reply within the MessageTimeout. Most are error scenarios. When something like this happens, the Charging Station SHALL retry the messages a couple of times. |
| Purpose | To verify if the Charging Station is able to resend TransactionEvent messages until the max retry count is reached, when the CSMS does not reply. |
| Prerequisite(s) | N/a |

Before (Preparations)

Configuration State:

MessageAttemptsTransactionEvent is <Configured message_attempts_transaction_event> (Must be > 1)

MessageAttemptIntervalTransactionEvent is <Configured message_attempts_transaction_event_interval>

Memory State:

N/a

Reusable State(s):

State is *Authorized*

State is *EVConnectedPreSession*

Main (Test scenario)

| Charging Station | CSMS |
|--|--|
| <u>Note(s)</u> : Step 1, 2, 3, & 4 are optional | |
| 1. The Charging Stations sends a TransactionEventRequest with: - triggerReason <i>SignedDataReceived</i> | 2. The Test System responds with a TransactionEventResponse |
| 3. The Charging Stations sends a TransactionEventRequest with: - triggerReason <i>ChargingStateChanged</i> - chargingState <i>SuspendedEVSE</i> | 4. The Test System responds with a TransactionEventResponse |
| <u>Note(s)</u> : Step 5 is repeated for the configured number of times | |
| 5. The Charging Stations sends a TransactionEventRequest | |

Tool validations

* Step 1:

- **triggerReason** *SignedDataReceived*

* Step 3:

- **triggerReason** *ChargingStateChanged*

- **chargingState** *SuspendedEVSE*

* Step 5:

- Needs to be sent a number of times equal to <Configured message_attempts_transaction_event> with an interval of (<Configured message_attempts_transaction_event_interval> * the number of preceding transmissions of this same message) + *OCPPCommCtrlr.MessageTimeout.Default*.

- The Test System waits an additional MessageAttemptsTransactionEvent iteration where the interval is multiplied again, to validate if the Charging Station stops resending the TransactionRequest message(s).

Post scenario validations:

N/a

TC_E_50_CS: Retry sending transaction message when failed - Max retry count reached - CallError

| | |
|-------------------|--|
| Test case name | Retry sending transaction message when failed - Max retry count reached - CallError |
| Test case Id | TC_E_50_CS |
| Use case Id(s) | E13 |
| Requirement(s) | E13.FR.01,E13.FR.02,E13.FR.03,E13.FR.04 |
| System under test | Charging Station |
| Description | There are situations/issues why a CSMS might not accept a transaction related message, or does not reply within the MessageTimeout. Most are error scenarios. When something like this happens, the Charging Station SHALL retry the messages a couple of times. |
| Purpose | To verify if the Charging Station is able to resend TransactionEvent messages until the max retry count is reached, when the CSMS does not reply. |
| Prerequisite(s) | N/a |

Before (Preparations)

Configuration State:

MessageAttemptsTransactionEvent is <Configured message_attempts_transaction_event> (Must be > 1)

MessageAttemptIntervalTransactionEvent is <Configured message_attempts_transaction_event_interval>

Memory State:

N/a

Reusable State(s):

State is *Authorized*

State is *EVConnectedPreSession*

Main (Test scenario)

| Charging Station | CSMS |
|--|--|
| <u>Note(s)</u> : Step 1, 2, 3, & 4 are optional | |
| 1. The Charging Stations sends a TransactionEventRequest with: - triggerReason <i>SignedDataReceived</i> | 2. The Test System responds with a TransactionEventResponse |
| 3. The Charging Stations sends a TransactionEventRequest with: - triggerReason <i>ChargingStateChanged</i> - chargingState <i>SuspendedEVSE</i> | 4. The Test System responds with a TransactionEventResponse |
| <u>Note(s)</u> : Step 5 is repeated for the configured number of times | |
| 5. The Charging Stations sends a TransactionEventRequest | 6. The Test System responds with a CallError with errorCode <i>InternalError</i> |

Tool validations

* Step 1:

- **triggerReason** *SignedDataReceived*

* Step 3:

- **triggerReason** *ChargingStateChanged*

- **chargingState** *SuspendedEVSE*

* Step 5:

- Needs to be sent a number of times equal to <Configured message_attempts_transaction_event> with an interval of the <Configured message_attempts_transaction_event_interval> multiplied by the number of preceding transmissions of this same message.

- The Test System waits an additional MessageAttemptsTransactionEvent iteration where the interval is multiplied again, to validate if the Charging Station stops resending the TransactionRequest message(s).

Post scenario validations:

N/a

TC_E_42_CS: Retry sending transaction message when failed - Success before reaching the max retry count

| | |
|--------------------------|--|
| Test case name | Retry sending transaction message when failed - Success before reaching the max retry count |
| Test case Id | TC_E_42_CS |
| Use case Id(s) | E13 |
| Requirement(s) | E13.FR.01,E13.FR.02,E13.FR.03 |
| System under test | Charging Station |
| Description | There are situations/issues why a CSMS might not accept a transaction related message, or does not reply within the MessageTimeout. Most are error scenarios. When something like this happens, the Charging Station SHALL retry the messages a couple of times. |
| Purpose | To verify if the Charging Station is able to resend TransactionEvent messages when the CSMS does not reply and stops resending after getting a response before the max retry count is reached. |
| Prerequisite(s) | N/a |

| |
|---|
| Before (Preparations) |
| Configuration State: MessageAttemptsTransactionEvent is <Configured message_attempts_transaction_event> (Must be > 2) MessageAttemptIntervalTransactionEvent is <Configured message_attempts_transaction_event_interval> |
| Memory State: N/a |
| Reusable State(s): State is <i>Authorized</i> State is <i>EVConnectedPreSession</i> |

| Main (Test scenario) | |
|--|--|
| Charging Station | CSMS |
| <u>Note(s)</u> : Step 1, 2, 3, & 4 are optional | |
| 1. The Charging Stations sends a TransactionEventRequest with: - triggerReason <i>SignedDataReceived</i> | 2. The Test System responds with a TransactionEventResponse |
| 3. The Charging Stations sends a TransactionEventRequest with: - triggerReason <i>ChargingStateChanged</i> - chargingState <i>SuspendedEVSE</i> | 4. The Test System responds with a TransactionEventResponse |
| <u>Note(s)</u> : The tool will ignore the first request and only respond to the second request | |
| 5. The Charging Stations sends a TransactionEventRequest | 6. The Test System responds with a TransactionEventResponse |

| |
|--|
| Tool validations |
| * Step 1: - triggerReason <i>SignedDataReceived</i> * Step 3: - triggerReason <i>ChargingStateChanged</i> - chargingState <i>SuspendedEVSE</i> * Step 5: - Needs to be sent 2 times with an interval of (<Configured message_attempts_transaction_event_interval> * the number of preceding transmissions of this same message) + <i>OcppCommCtrlr.MessageTimeout.Default</i> . - The Test System waits an additional MessageAttemptsTransactionEvent iteration where the interval is multiplied again, to validate if the Charging Station stops resending the TransactionRequest message(s). |
| Post scenario validations: N/a |

TC_E_51_CS: Retry sending transaction message when failed - Success before reaching the max retry count - CallError

| | |
|-------------------|--|
| Test case name | Retry sending transaction message when failed - Success before reaching the max retry count - CallError |
| Test case Id | TC_E_51_CS |
| Use case Id(s) | E13 |
| Requirement(s) | E13.FR.01,E13.FR.02,E13.FR.03 |
| System under test | Charging Station |
| Description | There are situations/issues why a CSMS might not accept a transaction related message, or does not reply within the MessageTimeout. Most are error scenarios. When something like this happens, the Charging Station SHALL retry the messages a couple of times. |
| Purpose | To verify if the Charging Station is able to resend TransactionEvent messages when the CSMS does not reply and stops resending after getting a response before the max retry count is reached. |
| Prerequisite(s) | N/a |

| |
|---|
| Before (Preparations) |
| Configuration State: MessageAttemptsTransactionEvent is <Configured message_attempts_transaction_event> (Must be > 2) MessageAttemptIntervalTransactionEvent is <Configured message_attempts_transaction_event_interval> |
| Memory State: N/a |
| Reusable State(s): State is <i>Authorized</i> State is <i>EVConnectedPreSession</i> |

| Main (Test scenario) | |
|---|--|
| Charging Station | CSMS |
| <u>Note(s)</u> : Step 1, 2, 3, & 4 are optional | |
| 1. The Charging Stations sends a TransactionEventRequest with: - triggerReason <i>SignedDataReceived</i> | 2. The Test System responds with a TransactionEventResponse |
| 3. The Charging Stations sends a TransactionEventRequest with: - triggerReason <i>ChargingStateChanged</i> - chargingState <i>SuspendedEVSE</i> | 4. The Test System responds with a TransactionEventResponse |
| <u>Note(s)</u> : The tool will send a <i>CallError</i> with <i>errorCode InternalError</i> to all requests except for the second request, there a <i>TransactionEventResponse</i> is send | |
| 5. The Charging Stations sends a TransactionEventRequest | 6. The Test System responds with a TransactionEventResponse |

| |
|--|
| Tool validations |
| * Step 1: - triggerReason <i>SignedDataReceived</i> * Step 3: - triggerReason <i>ChargingStateChanged</i> - chargingState <i>SuspendedEVSE</i> * Step 5: - Needs to be sent 2 times with an interval of (<Configured message_attempts_transaction_event_interval> * the number of preceding transmissions of this same message) + <i>OcppCommCtrlr.MessageTimeout.Default</i> . - The Test System waits an additional MessageAttemptsTransactionEvent iteration where the interval is multiplied again, to validate if the Charging Station stops resending the TransactionRequest message(s). |
| Post scenario validations: N/a |

TC_E_43_CS: Offline Behaviour - Transaction during offline period

| | |
|-------------------|--|
| Test case name | Offline Behaviour - Transaction during offline period |
| Test case Id | TC_E_43_CS |
| Use case Id(s) | E12 |
| Requirement(s) | E12.FR.01,E12.FR.02,E12.FR.06 |
| System under test | Charging Station |
| Description | The Charging Station queues TransactionEvent messages to inform the CSMS that a transaction occurred while the Charging Station was Offline. |
| Purpose | To verify if the Charging Station is able to queue TransactionEvent messages while it was offline. |
| Prerequisite(s) | The Charging Station supports authorization methods other than NoAuthorization |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>TransactionEventsInQueueEnded</i> |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Charging Stations sends a TransactionEventRequest | 2. The Test System responds with a TransactionEventResponse |
| <u>Note(s):</u> - The Charging Station will empty its Transaction message queue. This will contain one or more TransactionEventRequest messages | |

| |
|--|
| Tool validations |
| <p>* Step 1:</p> <p>All messages: TransactionEventRequest</p> <p>- offline must be <i>true</i></p> <p>One of the messages: TransactionEventRequest</p> <p>- eventType <i>Started</i></p> <p>One of the messages: TransactionEventRequest</p> <p>- eventType <i>Ended</i></p> |
| Post scenario validations: N/a |

NOTE | If the Charging Station supports ISO15118, this testcase needs to be executed using EIM.

TC_E_44_CS: Offline Behaviour - Stop transaction during offline period

| | |
|-------------------|---|
| Test case name | Offline Behaviour - Stop transaction during offline period |
| Test case Id | TC_E_44_CS |
| Use case Id(s) | E08 |
| Requirement(s) | E08.FR.01,E08.FR.04,E08.FR.05,E08.FR.06,E08.FR.07,E08.FR.08 |
| System under test | Charging Station |
| Description | The Charging Station queues TransactionEvent messages to inform the CSMS that a transaction occurred while the Charging Station was Offline. |
| Purpose | To verify if the Charging Station is able to queue TransactionEvent messages when the transaction stopped while the Charging Station was offline. |
| Prerequisite(s) | N/a |

| |
|---|
| Before (Preparations) Configuration State: OfflineThreshold is <Configured RetryBackOffWaitMinimum_duration> + 60.0 RetryBackOffWaitMinimum is <Configured RetryBackOffWaitMinimum_duration> RetryBackOffRandomRange is 0 <u>Note:</u> <Configured RetryBackOffWaitMinimum_duration should be long enough to execute manual tasks> |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| | 1. The Test System closes the WebSocket connection AND does not accept a reconnect. |
| <u>Manual Action:</u> Present the same idToken as used to start the transaction. | |
| <u>Manual Action:</u> Disconnect the EV and EVSE. | |
| <u>Manual Action:</u> Drive EV out of parking bay. (Only needed if TxStopPoint is ParkingBayOccupancy) | |
| | 2. The Test System accepts the reconnection attempt from the Charging Station. |
| 3. The Charging Stations sends a TransactionEventRequest | 4. The Test System responds with a TransactionEventResponse |
| <u>Note(s):</u> - The Charging Station will empty its Transaction message queue. This will contain one or more TransactionEventRequest messages | |

| |
|---|
| Tool validations |
| * Step 3: All messages: TransactionEventRequest - offline must be <i>true</i> One of the messages: TransactionEventRequest - eventType <i>Ended</i> |
| Post scenario validations: N/a |

NOTE | If the Charging Station supports ISO15118, this testcase needs to be executed using EIM.

TC_E_45_CS: Offline Behaviour - Stop transaction during offline period - Same GroupId

| | |
|-------------------|--|
| Test case name | Offline Behaviour - Stop transaction during offline period - Same GroupId |
| Test case Id | TC_E_45_CS |
| Use case Id(s) | E08 |
| Requirement(s) | E08.FR.02,E08.FR.04,E08.FR.05,E08.FR.06,E08.FR.07,E08.FR.08 |
| System under test | Charging Station |
| Description | The Charging Station queues TransactionEvent messages to inform the CSMS that a transaction occurred while the Charging Station was Offline. |
| Purpose | To verify if the Charging Station is able to queue TransactionEvent messages when the transaction stopped by an idToken with the same groupIdToken, while the Charging Station was offline. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The Charging Station supports Authorization cache OR Local Authorization List. - The Charging Station supports authorization methods other than NoAuthorization |

| |
|--|
| Before (Preparations) |
| <p>Configuration State:</p> <p>OfflineThreshold is <Configured RetryBackOffWaitMinimum_duration> + 60.0</p> <p>RetryBackOffWaitMinimum is <Configured RetryBackOffWaitMinimum_duration></p> <p>RetryBackOffRandomRange is 0</p> <p><u>Note:</u></p> <p><Configured RetryBackOffWaitMinimum_duration should be long enough to execute manual tasks></p> <p>Memory State:</p> <p>IdTokenCached for <Configured valid idtoken fields2> with <Configured GroupIdToken></p> <p>IdTokenLocalAuthList for <Configured valid idtoken fields2> with <Configured GroupIdToken></p> <p>Reusable State(s):</p> <p>State is Authorized with <Configured GroupIdToken></p> <p>Then proceed to state EnergyTransferStarted</p> |

| Main (Test scenario) | |
|--|---|
| Charging Station | CSMS |
| | 1. The Test System closes the WebSocket connection AND does not accept a reconnect. |
| <u>Manual Action:</u> Present <Configured valid idtoken fields2>. | |
| <u>Manual Action:</u> Disconnect the EV and EVSE. | |
| <u>Manual Action:</u> Drive EV out of parking bay. (Only needed if TxStopPoint is ParkingBayOccupancy) | |
| | 2. The Test System accepts the reconnection attempt from the Charging Station. |
| 3. The Charging Stations sends a TransactionEventRequest <u>Note(s):</u> - The Charging Station will empty its Transaction message queue. This will contain one or more TransactionEventRequest messages | 4. The Test System responds with a TransactionEventResponse |

| |
|--|
| Tool validations |
| <p>* Step 3:</p> <p>All messages: TransactionEventRequest</p> <ul style="list-style-type: none"> - offline must be true <p>One of the messages: TransactionEventRequest</p> <ul style="list-style-type: none"> - eventType Ended <p>Post scenario validations:</p> <p>N/a</p> |

NOTE | If the Charging Station supports ISO15118, this testcase needs to be executed using EIM.

TC_E_46_CS: End of charging process 15118

| | |
|-------------------|--|
| Test case name | End of charging process 15118 |
| Test case Id | TC_E_46_CS |
| Use case Id(s) | E15 |
| Requirement(s) | E15.FR.04, E15.FR.05 |
| System under test | Charging Station |
| Description | After receiving a SessionStopReq(Terminate) message from the EV, the Charging Station informs the CSMS that the authorization of the charging session has been stopped (by the EV). Depending on TxStopPoint this will also end the transaction. |
| Purpose | To verify whether the Charging Station is able to inform the CSMS that authorization of the charging session has been stopped (by the EV) and depending on TxStopPoint end the transaction. |
| Prerequisite(s) | N/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EVConnectedPreSession</i> State is <i>Authorized15118</i> State is <i>EnergyTransferStarted</i> |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Note:</u> The Charging Station receives a SessionStopReq(Terminate) message from the EV. | |
| 1. The Charging Station sends a TransactionEventRequest | 2. The Test System responds with a TransactionEventResponse |

| |
|--|
| Tool validations |
| <p>* Step 1:</p> <p>Message: TransactionEventRequest</p> <p>If <Configured TxStopPoint> contains "Authorized" or "PowerPathClosed" or "EnergyTransfer":</p> <ul style="list-style-type: none"> - eventType is <i>Ended</i> - triggerReason is <i>StopAuthorized</i> - transactionInfo.stoppedReason is <i>StoppedByEV</i> - transactionInfo.chargingState is <i>EVConnected</i> <p>If <Configured TxStopPoint> does not contain "Authorized" or "PowerPathClosed" or "EnergyTransfer":</p> <ul style="list-style-type: none"> - eventType is <i>Updated</i> - triggerReason = <i>StopAuthorized</i> - transactionInfo.chargingState is <i>EVConnected</i> |
| Post scenario validations: N/a |

F Remote Control

TC_F_01_CS: Remote start transaction - Cable plugin first

| | |
|-------------------|---|
| Test case name | Remote start transaction - Cable plugin first |
| Test case Id | TC_F_01_CS |
| Use case Id(s) | F01 |
| Requirement(s) | F01.FR.03, F01.FR.04, F01.FR.05, F01.FR.13, F01.FR.17, F01.FR.19, F02.FR.01 |
| System under test | Charging Station |
| Description | OCPP 2.x.x allows an EV driver to either first connect the EV and EVSE OR wait for/trigger a RequestStartTransactionRequest. Both sequences will result in being able to charge. |
| Purpose | To verify if the Charging Station is able to start a charging session when the EV driver first connects the EV and EVSE, before receiving a RequestStartTransactionRequest message. |
| Prerequisite(s) | - The Charging Station does NOT have a cable lock, which prevents the EV driver to connect the EV and EVSE before authorization. |

| |
|---|
| Before (Preparations) |
| Configuration State: AuthCtrlr.AuthEnabled is <i>true</i> (If implemented AND ReadWrite) AuthCtrlr.DisableRemoteAuthorization is <i>false</i> (If implemented) |
| Memory State: N/a |
| Reusable State(s): State is <i>EVConnectedPreSession</i> |

| | |
|--|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>Authorized</i> (remote) | |
| 2. Execute Reusable State <i>EnergyTransferStarted</i> | |

| |
|--|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_F_02_CS: Remote start transaction - Remote start first - AuthorizeRemoteStart is true

| | |
|-------------------|---|
| Test case name | Remote start transaction - Remote start first - AuthorizeRemoteStart is true |
| Test case Id | TC_F_02_CS |
| Use case Id(s) | F02 |
| Requirement(s) | F02.FR.01, F01.FR.01 |
| System under test | Charging Station |
| Description | OCPP 2.x.x allows an EV driver to either first wait for/trigger a RequestStartTransactionRequest OR connect the EV and EVSE. Both sequences will result in being able to charge. |
| Purpose | To verify if the Charging Station is able to start a charging session when the Charging Stations receives a RequestStartTransactionRequest message (while AuthorizeRemoteStart is true), before the EV driver connects the EV and EVSE (within the connectionTimeout). The Charging Station has to authorize beforehand like a local action to start a transaction. |
| Prerequisite(s) | - AuthEnabled is NOT implemented with mutability ReadOnly and the value set to false AND - AuthorizeRemoteStart is NOT implemented with mutability ReadOnly and the value set to false |

| |
|--|
| Before (Preparations) |
| Configuration State: AuthCtrlr.AuthEnabled is <i>true</i> (If implemented AND ReadWrite) AuthCtrlr.DisableRemoteAuthorization is <i>false</i> (If implemented) AuthorizeRemoteStart is <i>true</i> (If ReadWrite) |
| Memory State: N/a |
| Reusable State(s): State is <i>ParkingBayOccupied</i> (Optional state) |

| | |
|--|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>Authorized</i> (remote) | |
| 2. Execute Reusable State <i>EnergyTransferStarted</i> | |

| |
|--|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_F_03_CS: Remote start transaction - Remote start first - AuthorizeRemoteStart is false

| | |
|-------------------|--|
| Test case name | Remote start transaction - Remote start first - AuthorizeRemoteStart is false |
| Test case Id | TC_F_03_CS |
| Use case Id(s) | F02 |
| Requirement(s) | F02.FR.01, F01.FR.02 |
| System under test | Charging Station |
| Description | OCPP 2.x.x allows an EV driver to either first wait for/trigger a RequestStartTransactionRequest OR connect the EV and EVSE. Both sequences will result in being able to charge. |
| Purpose | To verify if the Charging Station is able to start a charging session when the Charging Stations receives a RequestStartTransactionRequest message (while AuthorizeRemoteStart is false), before the EV driver connects the EV and EVSE (within the connectionTimeout). The Charging station does NOT have to authorize beforehand like a local action to start a transaction. |
| Prerequisite(s) | AuthorizeRemoteStart is NOT implemented with mutability ReadOnly and the value set to true |

| |
|---|
| Before (Preparations) |
| Configuration State: AuthCtrlr.AuthEnabled is <i>true</i> (If implemented AND ReadWrite) AuthCtrlr.DisableRemoteAuthorization is <i>false</i> (If implemented) AuthorizeRemoteStart is <i>false</i> (If ReadWrite) |
| Memory State: N/a |
| Reusable State(s): State is <i>ParkingBayOccupied</i> (Optional state) |

| | |
|--|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>Authorized</i> (remote) | |
| 2. Execute Reusable State <i>EnergyTransferStarted</i> | |

| |
|--|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_F_04_CS: Remote start transaction - Remote start first - Cable plugin timeout

| | |
|-------------------|--|
| Test case name | Remote start transaction - Remote start first - Cable plugin timeout |
| Test case Id | TC_F_04_CS |
| Use case Id(s) | F02, E03 |
| Requirement(s) | F02.FR.01, E03.FR.01, E03.FR.05 |
| System under test | Charging Station |
| Description | OCPP 2.x.x allows an EV driver to either first wait for/trigger a RequestStartTransactionRequest OR connect the EV and EVSE. Both sequences will result in being able to charge. |
| Purpose | To verify if the Charging Station is able to deauthorize the transaction after the EVConnectionTimeout has been reached. |
| Prerequisite(s) | N/a |

Before (Preparations)

Configuration State:

- TxCtrlr.EVConnectionTimeOut is <Configured ev_connection_timeout>
- AuthCtrlr.AuthEnabled is true (If implemented AND ReadWrite)
- AuthCtrlr.DisableRemoteAuthorization is false (If implemented)
- TxCtrlr.TxStartPoint is ParkingBayOccupancy OR Authorized (If supported)

Memory State:

N/a

Reusable State(s):

State is *Authorized* (remote)

Main (Test scenario)

| Charging Station | CSMS |
|--|--|
| 1. The Charging Station sends a TransactionEventRequest <u>Note(s):</u> - This step needs to be executed after the <Configured ev_connection_timeout> expires, if the transaction has been started. So in the case TxStartPoint contains ParkingBayOccupancy OR Authorized | 2. The Test System responds with a TransactionEventResponse |
| <u>Note(s):</u> Optionally the Charging Station can send a StatusNotificationRequest or NotifyEventRequest with status Available | |
| 3. Execute Reusable State <i>Authorized</i> (remote) <u>Note(s):</u> - This step is executed to verify if the EVSE is actually ready to start another charging session. | |

Tool validations

* Step 1:

Message: **TransactionEventRequest**

- **triggerReason** must be EVConnectTimeout
- **eventType** must be Ended

Post scenario validations:

N/a

TC_F_05_CS: Remote unlock Connector - With ongoing transaction

| | |
|-------------------|---|
| Test case name | Remote unlock Connector - With ongoing transaction |
| Test case Id | TC_F_05_CS |
| Use case Id(s) | F05 |
| Requirement(s) | F05.FR.01, F05.FR.02 |
| System under test | Charging Station |
| Description | This test case describes how the CSMS can be requested to sent an UnlockConnectorRequest to the charging station. It sometimes happens that a connector of a Charging Station socket does not unlock correctly. This happens most of the time when there is tension on the charging cable. This means the driver cannot unplug his charging cable from the Charging Station. To help a driver, the CSO can send a UnlockConnectorRequest to the Charging Station. The Charging Station will then try to unlock the connector again. |
| Purpose | To verify if the Chargin Station is able to ignore the UnlockConnectorRequest whith an ongoing transaction as described at the OCPP specification. |
| Prerequisite(s) | The Charging Station has a connector lock. |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State:

Transaction is ongoing on <Configured Connector>
State is [EnergyTransferStarted](#)

Main (Test scenario)

| Charging Station | CSMS |
|--|---|
| 2. The Charging Station responds with a UnlockConnectorResponse | 1. The Test System sends a UnlockConnectorRequest with evseld <Configured evseld> connectorId <Configured connectorId> |

Tool validations

* Step 2:

Message **UnlockConnectorResponse**- **status** *OngoingAuthorizedTransaction*

Post scenario validations:

- N/a

TC_F_06_CS: Remote unlock Connector - Without ongoing transaction - Accepted

| | |
|-------------------|---|
| Test case name | Remote unlock Connector - Without ongoing transaction - Accepted |
| Test case Id | TC_F_06_CS |
| Use case Id(s) | F05 |
| Requirement(s) | F05.FR.01, F05.FR.04 |
| System under test | Charging Station |
| Description | This test case describes how the CSMS can be requested to send an UnlockConnectorRequest to the charging station. It sometimes happens that a connector of a Charging Station socket does not unlock correctly. This happens most of the time when there is tension on the charging cable. This means the driver cannot unplug his charging cable from the Charging Station. To help a driver, the CSO can send a UnlockConnectorRequest to the Charging Station. The Charging Station will then try to unlock the connector again. |
| Purpose | To verify if the Charging Station is able to successfully unlock a connector without ongoing transaction as described in the OCPP specification. |
| Prerequisite(s) | The Charging Station has a connector lock. |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State:

N/a

Main (Test scenario)

| Charging Station | CSMS |
|--|---|
| 2. The Charging Station responds with a UnlockConnectorResponse | 1. The Test System sends a UnlockConnectorRequest with evseld <Configured evseld> connectorId <Configured connectorId> |

Tool validations

* Step 2:

Message **UnlockConnectorResponse**- **status** *Unlocked*

Post scenario validations:

- N/a

TC_F_07_CS: Remote unlock Connector - Without ongoing transaction - No cable connected

| | |
|-------------------|---|
| Test case name | Remote unlock Connector - Without ongoing transaction - No cable connected |
| Test case Id | TC_F_07_CS |
| Use case Id(s) | F05 |
| Requirement(s) | F05.FR.01, F05.FR.06 |
| System under test | Charging Station |
| Description | This test case describes how the CSMS can be requested to send an UnlockConnectorRequest to the charging station. It sometimes happens that a connector of a Charging Station socket does not unlock correctly. This happens most of the time when there is tension on the charging cable. This means the driver cannot unplug his charging cable from the Charging Station. To help a driver, the CSO can send a UnlockConnectorRequest to the Charging Station. The Charging Station will then try to unlock the connector again. |
| Purpose | To verify if the Charging Station is able to perform the remote unlock connector mechanism and report the result without ongoing transaction while no cable is connected as described at the OCPP specification. |
| Prerequisite(s) | The Charging Station has a connector lock. |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: No cable connected at <Configured Connector> |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a UnlockConnectorResponse | 1. The Test System sends a UnlockConnectorRequest with evseld <Configured evseld> connectorId <Configured connectorId> |

| |
|--|
| Tool validations |
| * Step 2: Message UnlockConnectorResponse - status <i>Unlocked</i> |
| Post scenario validations: - N/a |

TC_F_08_CS: Remote stop transaction - Success

| | |
|-------------------|--|
| Test case name | Remote stop transaction - Success |
| Test case Id | TC_F_08_CS |
| Use case Id(s) | F03 |
| Requirement(s) | F03.FR.02, F03.FR.03, F03.FR.07, F03.FR.09 |
| System under test | Charging Station |
| Description | The CSMS is able to stop a charging session remotely by sending a RequestStopTransactionRequest to the Charging Station. |
| Purpose | To verify if the Charging Station is able to stop a charging session when it receives a RequestStopTransactionRequest message. |
| Prerequisite(s) | N/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> |

| | |
|--|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>StopAuthorized</i> (remote) | |

| |
|-----------------------------------|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_F_09_CS: Remote stop transaction - Rejected

| | |
|-------------------|--|
| Test case name | Remote stop transaction - Rejected |
| Test case Id | TC_F_09_CS |
| Use case Id(s) | F03 |
| Requirement(s) | F03.FR.08 |
| System under test | Charging Station |
| Description | The CSMS is able to stop a charging session remotely by sending a RequestStopTransactionRequest to the Charging Station. |
| Purpose | To verify if the Charging Station will reject a RequestStopTransactionRequest message, if it contains a transactionId that cannot be matched to an active transaction. |
| Prerequisite(s) | N/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a RequestStopTransactionResponse | 1. The Test System sends a RequestStopTransactionRequest with transactionId <Different transactionId than provided by the Charging Station in TransactionEventRequest > |

| |
|---|
| Tool validations |
| * Step 2: Message: RequestStopTransactionResponse - status must be Rejected |
| Post scenario validations: N/a |

TC_F_10_CS: Remote unlock Connector - Without ongoing transaction - UnknownConnector

| | |
|-------------------|---|
| Test case name | Remote unlock Connector - Without ongoing transaction - UnknownConnector |
| Test case Id | TC_F_10_CS |
| Use case Id(s) | F05 |
| Requirement(s) | F05.FR.03 |
| System under test | Charging Station |
| Description | This test case describes how the CSMS can be requested to send an UnlockConnectorRequest to the charging station. It sometimes happens that a connector of a Charging Station socket does not unlock correctly. This happens most of the time when there is tension on the charging cable. This means the driver cannot unplug his charging cable from the Charging Station. To help a driver, the CSO can send a UnlockConnectorRequest to the Charging Station. The Charging Station will then try to unlock the connector again. |
| Purpose | To verify if the Charging Station is able to respond with a UnlockConnectorRequest with status <i>UnknownConnector</i> when the requested connector is unknown as described in the OCPP specification. |
| Prerequisite(s) | The Charging Station has a connector lock. |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State:

N/a

Main (Test scenario)

| Charging Station | CSMS |
|--|--|
| 2. The Charging Station responds with a UnlockConnectorResponse | 1. The Test System sends a UnlockConnectorRequest with evseld <Configured evseld> connectorId 999 |

Tool validations

* Step 2:

Message **UnlockConnectorResponse**- **status** *UnknownConnector*

Post scenario validations:

- N/a

TC_F_11_CS: Trigger message - MeterValues - Specific EVSE

| | |
|-------------------|--|
| Test case name | Trigger message - MeterValues - Specific EVSE |
| Test case Id | TC_F_11_CS |
| Use case Id(s) | F06 |
| Requirement(s) | F06.FR.02,F06.FR.04,F06.FR.05,F06.FR.06,F06.FR.10 |
| System under test | Charging Station |
| Description | The CSMS can request a Charging Station to send Charging Station-initiated messages. In the request the CSMS indicates which message it wishes to receive. |
| Purpose | To verify if the Charging Station is able to send a MeterValuesRequest message for a specific EVSE, after receiving a TriggerMessageRequest message. |
| Prerequisite(s) | The Charging Station supports sending MeterValues triggered by a TriggerMessageRequest. |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State(s):

N/a

Main (Test scenario)

| Charging Station | CSMS |
|---|--|
| 2. The Charging Station responds with a TriggerMessageResponse | 1. The Test System sends a TriggerMessageRequest With requestedMessage <i>MeterValues</i> evse.id <Configured evseld> |
| 3. The Charging Station sends a MeterValuesRequest | 4. The Test System responds with a MeterValuesResponse |

Tool validations

* Step 2:

Message: **TriggerMessageResponse**- **status** must be *Accepted*

* Step 3:

Message: **MeterValuesRequest**- **evseld** must be <Configured evseld>- **meterValue[0].sampledValue[0].context** must be *Trigger*

Post scenario validations:

N/a

TC_F_12_CS: Trigger message - MeterValues - All EVSE

| | |
|-------------------|--|
| Test case name | Trigger message - MeterValues - All EVSE |
| Test case Id | TC_F_12_CS |
| Use case Id(s) | F06 |
| Requirement(s) | F06.FR.04,F06.FR.05,F06.FR.06,F06.FR.10,F06.FR.11 |
| System under test | Charging Station |
| Description | The CSMS can request a Charging Station to send Charging Station-initiated messages. In the request the CSMS indicates which message it wishes to receive. |
| Purpose | To verify if the Charging Station is able to send a MeterValuesRequest message for all EVSE, after receiving a TriggerMessageRequest message. |
| Prerequisite(s) | The Charging Station supports sending MeterValues triggered by a TriggerMessageRequest. |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a TriggerMessageResponse | 1. The Test System sends a TriggerMessageRequest With requestedMessage <i>MeterValues</i> evse is omitted |
| 3. The Charging Station sends a MeterValuesRequest | 4. The Test System responds with a MeterValuesResponse |
| Note(s): - This step needs to be executed for every EVSE. | |

| |
|--|
| Tool validations |
| * Step 2: Message: TriggerMessageResponse - status must be <i>Accepted</i> * Step 3: Message: MeterValuesRequest - meterValue[0].sampledValue[0].context must be <i>Trigger</i> |
| Post scenario validations: N/a |

TC_F_13_CS: Trigger message - TransactionEvent - Specific EVSE

| | |
|-------------------|--|
| Test case name | Trigger message - TransactionEvent - Specific EVSE |
| Test case Id | TC_F_13_CS |
| Use case Id(s) | F06 |
| Requirement(s) | F06.FR.02,F06.FR.04,F06.FR.05,F06.FR.07,F06.FR.10 |
| System under test | Charging Station |
| Description | The CSMS can request a Charging Station to send Charging Station-initiated messages. In the request the CSMS indicates which message it wishes to receive. |
| Purpose | To verify if the Charging Station is able to send a TransactionEventRequest message for a specific EVSE, after receiving a TriggerMessageRequest message. |
| Prerequisite(s) | The Charging Station supports sending TransactionEvents triggered by a TriggerMessageRequest. |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> |

| Main (Test scenario) | |
|---|---|
| Charging Station | CSMS |
| 2. The Charging Station responds with a TriggerMessageResponse | 1. The Test System sends a TriggerMessageRequest With requestedMessage <i>TransactionEvent</i> evse.id <Configured evseld> |
| 3. The Charging Station sends a TransactionEventRequest | 4. The Test System responds with a TransactionEventResponse |

| |
|--|
| Tool validations |
| <p>* Step 2: Message: TriggerMessageResponse - status must be <i>Accepted</i></p> <p>* Step 3: Message: TransactionEventRequest - evse.id must be <i>omitted</i> or <Configured evseld> - triggerReason must be <i>Trigger</i> - transactionInfo.chargingState must be <i>Charging</i> - meterValue must be present - meterValue[0].sampledValue[0].context must be <i>Trigger</i></p> <p>Post scenario validations: N/a</p> |

TC_F_14_CS: Trigger message - TransactionEvent - All EVSE

| | |
|-------------------|--|
| Test case name | Trigger message - TransactionEvent - All EVSE |
| Test case Id | TC_F_14_CS |
| Use case Id(s) | F06 |
| Requirement(s) | F06.FR.04,F06.FR.05,F06.FR.07,F06.FR.10,F06.FR.11 |
| System under test | Charging Station |
| Description | The CSMS can request a Charging Station to send Charging Station-initiated messages. In the request the CSMS indicates which message it wishes to receive. |
| Purpose | To verify if the Charging Station is able to send a TransactionEventRequest message for all EVSE, after receiving a TriggerMessageRequest message. |
| Prerequisite(s) | The Charging Station supports sending TransactionEvents triggered by a TriggerMessageRequest. |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> for all EVSE |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a TriggerMessageResponse | 1. The Test System sends a TriggerMessageRequest With requestedMessage <i>TransactionEvent</i> evse is omitted |
| 3. The Charging Station sends a TransactionEventRequest | 4. The Test System responds with a TransactionEventResponse |
| Note(s): - This step needs to be executed for every EVSE. | |

| |
|--|
| Tool validations |
| * Step 2: Message: TriggerMessageResponse - status must be <i>Accepted</i> * Step 3: Message: TransactionEventRequest - evse.id must be <i><Configured evseld></i> - triggerReason must be <i>Trigger</i> - transactionInfo.chargingState must be <i>Charging</i> - meterValue must be present - meterValue[0].sampledValue[0].context must be <i>Trigger</i> |
| Post scenario validations: N/a |

TC_F_15_CS: Trigger message - LogStatusNotification - Idle

| | |
|--------------------------|--|
| Test case name | Trigger message - LogStatusNotification - Idle |
| Test case Id | TC_F_15_CS |
| Use case Id(s) | F06 |
| Requirement(s) | F06.FR.04,F06.FR.05,F06.FR.10,F06.FR.15 |
| System under test | Charging Station |
| Description | The CSMS can request a Charging Station to send Charging Station-initiated messages. In the request the CSMS indicates which message it wishes to receive. |
| Purpose | To verify if the Charging Station is able to send a LogStatusNotificationRequest with status Idle, after receiving a TriggerMessageRequest while NOT uploading a log file. |
| Prerequisite(s) | The Charging Station supports sending LogStatusNotifications triggered by a TriggerMessageRequest. |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a TriggerMessageResponse | 1. The Test System sends a TriggerMessageRequest With requestedMessage <i>LogStatusNotification</i> |
| 3. The Charging Station sends a LogStatusNotificationRequest | 4. The Test System responds with a LogStatusNotificationResponse |

| |
|---|
| Tool validations |
| <p>* Step 2: Message: TriggerMessageResponse - status must be <i>Accepted</i></p> <p>* Step 3: Message: LogStatusNotificationRequest - status must be <i>Idle</i></p> |
| Post scenario validations: N/a |

TC_F_16_CS: Trigger message - LogStatusNotification - Uploading

| | |
|-------------------|---|
| Test case name | Trigger message - LogStatusNotification - Uploading |
| Test case Id | TC_F_16_CS |
| Use case Id(s) | F06 |
| Requirement(s) | F06.FR.04,F06.FR.05,F06.FR.10,F06.FR.14 |
| System under test | Charging Station |
| Description | The CSMS can request a Charging Station to send Charging Station-initiated messages. In the request the CSMS indicates which message it wishes to receive. |
| Purpose | To verify if the Charging Station is able to send a LogStatusNotificationRequest with status Uploading, after receiving a TriggerMessageRequest while uploading a log file. |
| Prerequisite(s) | The Charging Station supports sending LogStatusNotifications triggered by a TriggerMessageRequest. |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State(s):

N/a

Main (Test scenario)

| Charging Station | CSMS |
|---|--|
| 2. The Charging Station responds with a GetLogResponse | 1. The Test System sends a GetLogRequest With logType <i>DiagnosticsLog</i> log.remoteLocation is <Configured log_location> |
| 3. The Charging Station sends a LogStatusNotificationRequest | 4. The Test System responds with a LogStatusNotificationResponse |
| 6. The Charging Station responds with a TriggerMessageResponse | 5. The Test System sends a TriggerMessageRequest With requestedMessage <i>LogStatusNotification</i> |
| 7. The Charging Station sends a LogStatusNotificationRequest | 8. The Test System responds with a LogStatusNotificationResponse |

Tool validations

* Step 2:

Message: **GetLogResponse**- **status** must be *Accepted*

* Step 3:

Message: **LogStatusNotificationRequest**- **status** must be *Uploading*

* Step 6:

Message: **TriggerMessageResponse**- **status** must be *Accepted*

* Step 7:

Message: **LogStatusNotificationRequest**- **status** must be *Uploading*

Post scenario validations:

N/a

TC_F_17_CS: Trigger message - FirmwareStatusNotification - Specific EVSE not relevant

| | |
|--------------------------|--|
| Test case name | Trigger message - FirmwareStatusNotification - Specific EVSE not relevant |
| Test case Id | TC_F_17_CS |
| Use case Id(s) | F06 |
| Requirement(s) | F06.FR.03,F06.FR.04,F06.FR.05,F06.FR.10 |
| System under test | Charging Station |
| Description | The CSMS can request a Charging Station to send Charging Station-initiated messages. In the request the CSMS indicates which message it wishes to receive. |
| Purpose | To verify if the Charging Station is able to send a FirmwareStatusNotificationRequest, after receiving a TriggerMessageRequest even when the CSMS an evseld which is not relevant for the requestedMessage FirmwareStatusNotification. |
| Prerequisite(s) | The Charging Station supports sending FirmwareStatusNotifications triggered by a TriggerMessageRequest. |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a TriggerMessageResponse | 1. The Test System sends a TriggerMessageRequest With requestedMessage <i>FirmwareStatusNotification</i> evse.id is <Configured evseld> |
| 3. The Charging Station sends a FirmwareStatusNotificationRequest | 4. The Test System responds with a FirmwareStatusNotificationResponse |

| |
|--|
| Tool validations |
| <p>* Step 2: Message: TriggerMessageResponse - status must be <i>Accepted</i></p> <p>* Step 3: Message: FirmwareStatusNotificationRequest - status must be <i>Idle</i></p> |
| Post scenario validations: N/a |

TC_F_18_CS: Trigger message - FirmwareStatusNotification - Idle

| | |
|-------------------|--|
| Test case name | Trigger message - FirmwareStatusNotification - Idle |
| Test case Id | TC_F_18_CS |
| Use case Id(s) | F06 |
| Requirement(s) | F06.FR.04,F06.FR.05,F06.FR.10,F06.FR.16,L01.FR.25 |
| System under test | Charging Station |
| Description | The CSMS can request a Charging Station to send Charging Station-initiated messages. In the request the CSMS indicates which message it wishes to receive. |
| Purpose | To verify if the Charging Station is able to send a FirmwareStatusNotificationRequest with status Idle, after receiving a TriggerMessageRequest while NOT downloading a firmware file. |
| Prerequisite(s) | The Charging Station supports sending FirmwareStatusNotifications triggered by a TriggerMessageRequest. |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|--|---|
| Charging Station | CSMS |
| 2. The Charging Station responds with a TriggerMessageResponse | 1. The Test System sends a TriggerMessageRequest With requestedMessage <i>FirmwareStatusNotification</i> |
| 3. The Charging Station sends a FirmwareStatusNotificationRequest | 4. The Test System responds with a FirmwareStatusNotificationResponse |

| |
|--|
| Tool validations |
| <p>* Step 2: Message: TriggerMessageResponse - status must be <i>Accepted</i></p> <p>* Step 3: Message: FirmwareStatusNotificationRequest - status must be <i>Idle</i></p> |
| Post scenario validations: N/a |

TC_F_19_CS: Trigger message - FirmwareStatusNotification - Downloading

| | |
|-------------------|---|
| Test case name | Trigger message - FirmwareStatusNotification - Downloading |
| Test case Id | TC_F_19_CS |
| Use case Id(s) | F06 |
| Requirement(s) | F06.FR.04,F06.FR.05,F06.FR.10,L01.FR.26 |
| System under test | Charging Station |
| Description | The CSMS can request a Charging Station to send Charging Station-initiated messages. In the request the CSMS indicates which message it wishes to receive. |
| Purpose | To verify if the Charging Station is able to send a FirmwareStatusNotificationRequest with status Downloading, after receiving a TriggerMessageRequest while downloading a firmware file. |
| Prerequisite(s) | The Charging Station supports sending FirmwareStatusNotifications triggered by a TriggerMessageRequest. |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a UpdateFirmwareResponse | 1. The Test System sends a UpdateFirmwareRequest firmware.location is <Configured firmware_location> firmware.retrieveDateTime is <Current dateTime - 2 hours> firmware.installDateTime is omitted firmware.signingCertificate is <Configured signingCertificate> firmware.signature is <Configured invalid firmware signature> |
| 3. The Charging Station sends a FirmwareStatusNotificationRequest | 4. The Test System responds with a FirmwareStatusNotificationResponse |
| 6. The Charging Station responds with a TriggerMessageResponse | 5. The Test System sends a TriggerMessageRequest With requestedMessage <i>FirmwareStatusNotification</i> |
| 7. The Charging Station sends a FirmwareStatusNotificationRequest | 8. The Test System responds with a FirmwareStatusNotificationResponse |
| <i>Note: Step 9 through 14 are cleanup to prevent an ongoing firmware update after the testcase is already ended. The behavior part of these steps is part of TC_L_06_CS and therefore not part of the scope for this testcase.</i> | |
| 9. The Charging Station sends a FirmwareStatusNotificationRequest . With status Downloaded | 10. The Test System responds with a FirmwareStatusNotificationResponse . |
| 11. The Charging Station sends a FirmwareStatusNotificationRequest . With status InvalidSignature | 12. The Test System responds with a FirmwareStatusNotificationResponse . |
| 13. The Charging Station sends a SecurityEventNotificationRequest . With type InvalidFirmwareSignature | 14. The Test System responds with a SecurityEventNotificationResponse . |

| Tool validations |
|---|
| <p>* Step 2: Message: UpdateFirmwareResponse - status must be <i>Accepted</i></p> <p>* Step 3: Message: FirmwareStatusNotificationRequest - status must be <i>Downloading</i></p> <p>* Step 6: Message: TriggerMessageResponse - status must be <i>Accepted</i></p> <p>* Step 7: Message: FirmwareStatusNotificationRequest - status must be <i>Downloading</i></p> |
| <p>Post scenario validations: N/a</p> |

TC_F_20_CS: Trigger message - Heartbeat

| | |
|--------------------------|--|
| Test case name | Trigger message - Heartbeat |
| Test case Id | TC_F_20_CS |
| Use case Id(s) | F06 |
| Requirement(s) | F06.FR.04,F06.FR.05,F06.FR.10 |
| System under test | Charging Station |
| Description | The CSMS can request a Charging Station to send Charging Station-initiated messages. In the request the CSMS indicates which message it wishes to receive. |
| Purpose | To verify if the Charging Station is able to send a HeartbeatRequest, after receiving a TriggerMessageRequest. |
| Prerequisite(s) | The Charging Station supports sending Heartbeats triggered by a TriggerMessageRequest. |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a TriggerMessageResponse | 1. The Test System sends a TriggerMessageRequest With requestedMessage Heartbeat |
| 3. The Charging Station sends a HeartbeatRequest | 4. The Test System responds with a HeartbeatResponse |

| |
|--|
| Tool validations |
| * Step 2: Message: TriggerMessageResponse - status must be <i>Accepted</i> |
| Post scenario validations: N/a |

TC_F_23_CS: Trigger message - StatusNotification - Specific EVSE - Available

| | |
|-------------------|---|
| Test case name | Trigger message - StatusNotification - Specific EVSE - Available |
| Test case Id | TC_F_23_CS |
| Use case Id(s) | F06 |
| Requirement(s) | F06.FR.02,F06.FR.04,F06.FR.05,F06.FR.10 |
| System under test | Charging Station |
| Description | The CSMS can request a Charging Station to send Charging Station-initiated messages. In the request the CSMS indicates which message it wishes to receive. |
| Purpose | To verify if the Charging Station is able to send a StatusNotificationRequest message for a specific available EVSE/Connector, after receiving a TriggerMessageRequest message. |
| Prerequisite(s) | The Charging Station supports sending StatusNotifications triggered by a TriggerMessageRequest. |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|---|---|
| Charging Station | CSMS |
| 2. The Charging Station responds with a TriggerMessageResponse | 1. The Test System sends a TriggerMessageRequest With requestedMessage <i>StatusNotification</i> evse.id <Configured evseId> evse.connectorId <Configured connectorId> |
| 3. The Charging Station notifies the CSMS about the current state of the connector. | 4. The Test System responds accordingly. |

| |
|--|
| Tool validations |
| <p>* Step 2: Message: TriggerMessageResponse - status must be <i>Accepted</i></p> <p>* Step 3: Message: StatusNotificationRequest - connectorStatus <i>Available</i> Message: NotifyEventRequest - eventData[0].trigger <i>Delta</i> - eventData[0].actualValue <i>"Available"</i> - eventData[0].component.name <i>"Connector"</i> - eventData[0].variable.name <i>"AvailabilityState"</i></p> |
| Post scenario validations: N/a |

TC_F_24_CS: Trigger message - StatusNotification - Specific EVSE - Occupied

| | |
|-------------------|--|
| Test case name | Trigger message - StatusNotification - Specific EVSE - Occupied |
| Test case Id | TC_F_24_CS |
| Use case Id(s) | F06 |
| Requirement(s) | F06.FR.02,F06.FR.04,F06.FR.05,F06.FR.10 |
| System under test | Charging Station |
| Description | The CSMS can request a Charging Station to send Charging Station-initiated messages. In the request the CSMS indicates which message it wishes to receive. |
| Purpose | To verify if the Charging Station is able to send a StatusNotificationRequest message for a specific occupied EVSE/Connector, after receiving a TriggerMessageRequest message. |
| Prerequisite(s) | The Charging Station supports sending StatusNotifications triggered by a TriggerMessageRequest. |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EVConnectedPreSession</i> |

| Main (Test scenario) | |
|---|---|
| Charging Station | CSMS |
| 2. The Charging Station responds with a TriggerMessageResponse | 1. The Test System sends a TriggerMessageRequest With requestedMessage <i>StatusNotification</i> evse.id <Configured evseId> evse.connectorId <Configured connectorId> |
| 3. The Charging Station notifies the CSMS about the current state of the connector. | 4. The Test System responds accordingly. |

| |
|--|
| Tool validations |
| <p>* Step 2: Message: TriggerMessageResponse - status must be <i>Accepted</i></p> <p>* Step 3: Message: StatusNotificationRequest - connectorStatus <i>Occupied</i> Message: NotifyEventRequest - eventData[0].trigger <i>Delta</i> - eventData[0].actualValue <i>"Occupied"</i> - eventData[0].component.name <i>"Connector"</i> - eventData[0].variable.name <i>"AvailabilityState"</i></p> |
| Post scenario validations: N/a |

TC_F_26_CS: Trigger message - BootNotification - Rejected

| | |
|-------------------|---|
| Test case name | Trigger message - BootNotification - Rejected |
| Test case Id | TC_F_26_CS |
| Use case Id(s) | F06 |
| Requirement(s) | F06.FR.04,F06.FR.05,F06.FR.17 |
| System under test | Charging Station |
| Description | The CSMS can request a Charging Station to send Charging Station-initiated messages. In the request the CSMS indicates which message it wishes to receive. |
| Purpose | To verify if the Charging Station rejects resending a BootNotificationRequest, when it has already received an accepted on a previously sent BootNotification, after receiving a TriggerMessageRequest. |
| Prerequisite(s) | The Charging Station supports sending BootNotification triggered by a TriggerMessageRequest. |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a TriggerMessageResponse | 1. The Test System sends a TriggerMessageRequest With requestedMessage <i>BootNotification</i> |

| |
|--|
| Tool validations |
| * Step 2: Message: TriggerMessageResponse - status must be <i>Rejected</i> |
| Post scenario validations: N/a |

TC_F_27_CS: Trigger message - NotImplemented

| | |
|--------------------------|---|
| Test case name | Trigger message - NotImplemented |
| Test case Id | TC_F_27_CS |
| Use case Id(s) | F06 |
| Requirement(s) | F06.FR.08 |
| System under test | Charging Station |
| Description | The CSMS can request a Charging Station to send Charging Station-initiated messages. In the request the CSMS indicates which message it wishes to receive. |
| Purpose | To verify if the Charging Station is able to report it has not implemented sending a SignCombinedCertificateRequest, after receiving a TriggerMessageRequest. |
| Prerequisite(s) | The Charging Station does NOT support sending SignCombinedCertificates triggered by a TriggerMessageRequest. |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a TriggerMessageResponse | 1. The Test System sends a TriggerMessageRequest With requestedMessage <i>SignCombinedCertificate</i> |

| |
|--|
| Tool validations |
| * Step 2: Message: TriggerMessageResponse - status must be <i>NotImplemented</i> |
| Post scenario validations: N/a |

G Availability**TC_G_01_CS: Connector status Notification - Available to Occupied**

| | |
|--------------------------|--|
| Test case name | Connector status Notification - Available to Occupied |
| Test case Id | TC_G_01_CS |
| Use case Id(s) | G01, N07 |
| Requirement(s) | G01.FR.01, N07.FR.19 |
| System under test | Charging Station |
| Description | A Charging Station sends a notification to the CSMS to inform the CSMS about a Connector status change. This can be done in two ways. Via a StatusNotificationRequest or a NotifyEventRequest from the device model. |
| Purpose | To verify whether the Charging Station is able to report that its connector is <i>Occupied</i> . |
| Prerequisite(s) | N/a |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>EVConnectedPreSession</i> | |

| |
|--|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_G_02_CS: Connector status Notification - Occupied to Available

| | |
|-------------------|--|
| Test case name | Connector status Notification - Occupied to Available |
| Test case Id | TC_G_02_CS |
| Use case Id(s) | G01, N07 |
| Requirement(s) | G01.FR.01, N07.FR.19 |
| System under test | Charging Station |
| Description | A Charging Station sends a notification to the CSMS to inform the CSMS about a Connector status change. This can be done in two ways. Via a StatusNotificationRequest or a NotifyEventRequest from the device model. |
| Purpose | To verify whether the Charging Station is able to report that its connector is Available_. |
| Prerequisite(s) | N/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EVConnectedPreSession</i> |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action</u> : <i>Disconnect the EV and EVSE.</i> | |
| 3. The Charging Station notifies the CSMS about the current state of the connector. | 4. The Test System responds accordingly. |

| |
|---|
| Tool validations |
| <p>* Step 3:</p> <p>Message: StatusNotificationRequest</p> <ul style="list-style-type: none"> - connectorStatus <i>Available</i> <p>Message: NotifyEventRequest</p> <ul style="list-style-type: none"> - eventData[0].trigger <i>Delta</i> - eventData[0].actualValue <i>"Available"</i> - eventData[0].component.name <i>"Connector"</i> - eventData[0].variable.name <i>"AvailabilityState"</i> |
| Post scenario validations: N/a |

TC_G_03_CS: Change Availability EVSE - Operative to inoperative

| | |
|--------------------------|--|
| Test case name | Change Availability EVSE - Operative to inoperative |
| Test case Id | TC_G_03_CS |
| Use case Id(s) | G03 |
| Requirement(s) | G03.FR.01, G03.FR.02, G03.FR.04, G03.FR.06 |
| System under test | Charging Station |
| Description | This test case covers how the CSMS requests the Charging Station to change the availability of one of the EVSEs from Operative to Inoperative. An EVSE is considered Operative in any status other than Faulted and Unavailable. |
| Purpose | To verify if the Charging Station is able to perform the change availability mechanism as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>Unavailable</i> for <Configured evseld> | |

| |
|---|
| Tool validations |
| N/a |
| Post scenario validations: - A message to report the state of a connector has been received for all connectors belonging to the specified EVSE. |

TC_G_04_CS: Change Availability EVSE - Inoperative to operative

| | |
|-------------------|--|
| Test case name | Change Availability EVSE - Inoperative to operative |
| Test case Id | TC_G_04_CS |
| Use case Id(s) | G03 |
| Requirement(s) | G03.FR.01, G03.FR.02, G03.FR.04, G03.FR.07 |
| System under test | Charging Station |
| Description | This test case covers how the CSMS requests the Charging Station to change the availability of one of the EVSEs from Inoperative to Operative. An EVSE is considered Operative in any status other than Faulted and Unavailable. |
| Purpose | To verify if the Charging Station is able to perform the change availability mechanism as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: <i>Unavailable</i> for <Configured evseId> |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|---|--|
| Charging Station | CSMS |
| 2. The Charging Station responds with a ChangeAvailabilityResponse | 1. The Test System sends a ChangeAvailabilityRequest with operationalStatus <i>Operative</i> and evse.id <i><Configured evseId></i> |
| 3. The Charging Station notifies the CSMS about the current state of all connectors belonging to the specified EVSE (and optionally also from the EVSE itself). | 4. The Test System responds accordingly. |

| |
|---|
| Tool validations |
| <p>* Step 2: Message ChangeAvailabilityResponse - status <i>Accepted</i></p> <p>* Step 3: Message: StatusNotificationRequest - connectorStatus <i>Available</i> - evseId <Configured evseId></p> <p>Message: NotifyEventRequest - eventData[0].trigger <i>Delta</i> - eventData[0].actualValue <i>"Available"</i> - eventData[0].component.name <i>"EVSE" / Connector</i> - eventData[0].component.evse.id <Configured evseId> - eventData[0].variable.name <i>"AvailabilityState"</i></p> |
| <p>Post scenario validations: - A message to report the state of a connector has been received for all connectors belonging to the specified EVSE.</p> |

TC_G_05_CS: Change Availability Charging Station - Operative to inoperative

| | |
|--------------------------|--|
| Test case name | Change Availability Charging Station - Operative to inoperative |
| Test case Id | TC_G_05_CS |
| Use case Id(s) | G04 |
| Requirement(s) | G04.FR.01, G04.FR.02, G04.FR.03, G04.FR.05, G04.FR.07 |
| System under test | Charging Station |
| Description | <p>This test case describes how the CSMS requests the Charging Station to change the availability from operative to inoperative.</p> <p>A Charging Station is considered Operative when it is charging or ready for charging.</p> <p>A Charging Station is considered Inoperative when it does not allow any charging.</p> |
| Purpose | To verify if the Charging Station is able to perform the change availability mechanism as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>Unavailable</i> | |

| |
|---|
| Tool validations |
| N/a |
| Post scenario validations: - A message to report the state of a connector has been received for all connectors. |

TC_G_06_CS: Change Availability Charging Station - Inoperative to operative

| | |
|-------------------|--|
| Test case name | Change Availability Charging Station - Inoperative to operative |
| Test case Id | TC_G_06_CS |
| Use case Id(s) | G04 |
| Requirement(s) | G04.FR.01, G04.FR.02, G04.FR.03, G04.FR.05, G04.FR.08 |
| System under test | Charging Station |
| Description | <p>This test case describes how the CSMS requests the Charging Station to change the availability from inoperative to operative.</p> <p>A Charging Station is considered Operative when it is charging or ready for charging.</p> <p>A Charging Station is considered Inoperative when it does not allow any charging.</p> |
| Purpose | To verify if the Charging Station is able to perform the change availability mechanism as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>Unavailable</i> |

| Main (Test scenario) | |
|---|--|
| Charging Station | CSMS |
| 2. The Charging Station responds with a ChangeAvailabilityResponse | 1. The Test System sends a ChangeAvailabilityRequest with operationalStatus <i>Operative</i> |
| 3. The Charging Station notifies the CSMS about the current state of all its connectors (and optionally also the Charging Station itself and all EVSE). | 4. The Test System responds accordingly. |

| |
|---|
| Tool validations |
| <p>* Step 2: Message ChangeAvailabilityResponse - status <i>Accepted</i></p> <p>* Step 3: Message: StatusNotificationRequest - connectorStatus <i>Available</i> Message: NotifyEventRequest - eventData[0].trigger <i>Delta</i> - eventData[0].actualValue <i>"Available"</i> - eventData[0].component.name <i>"ChargingStation" / EVSE / Connector</i> - eventData[0].variable.name <i>"AvailabilityState"</i></p> |
| Post scenario validations: - A message to report the state of a connector has been received for all connectors. |

TC_G_07_CS: Change Availability Connector - Operative to inoperative

| | |
|-------------------|---|
| Test case name | Change Availability Connector - Operative to inoperative |
| Test case Id | TC_G_07_CS |
| Use case Id(s) | G03 |
| Requirement(s) | G03.FR.01, G03.FR.02, G03.FR.04, G03.FR.06 |
| System under test | Charging Station |
| Description | This test case covers how the CSMS requests the Charging Station to change the availability of one of the Connectors of one EVSE from Operative to Inoperative. A Connector is considered Operative in any status other than Faulted and Unavailable. |
| Purpose | To verify if the Charging Station is able to perform the change availability mechanism as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>Unavailable</i> for <Configured connectorId> | |

| |
|---|
| Tool validations |
| N/a |
| Post scenario validations: - A message to report the state of the connector has been received. |

TC_G_08_CS: Change Availability Connector - Inoperative to operative

| | |
|-------------------|---|
| Test case name | Change Availability Connector - Inoperative to operative |
| Test case Id | TC_G_08_CS |
| Use case Id(s) | G03 |
| Requirement(s) | G03.FR.01, G03.FR.02, G03.FR.04, G03.FR.07 |
| System under test | Charging Station |
| Description | This test case covers how the CSMS requests the Charging Station to change the availability of one of the Connectors from one EVSE from Inoperative to Operative. A Connector is considered Operative in any status other than Faulted and Unavailable. |
| Purpose | To verify if the Charging Station is able to perform the change availability mechanism as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: <i>Unavailable</i> for <Configured connectorId> |
| Reusable State(s): N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a ChangeAvailabilityResponse | 1. The Test System sends a ChangeAvailabilityRequest with operationalStatus <i>Operative</i> and evse.id <Configured evseld> and evse.connectorId <Configured connectorId> |
| 3. The Charging Station notifies the CSMS about the current state of the connectors. | 4. The Test System responds accordingly. |

| |
|---|
| Tool validations |
| * Step 2: Message ChangeAvailabilityResponse - status <i>Accepted</i> * Step 3: Message: StatusNotificationRequest - connectorStatus <i>Available</i> - evseld <Configured evseld> - connectorId <Configured connectorId> Message: NotifyEventRequest - eventData[0].trigger <i>Delta</i> - eventData[0].actualValue <i>"Available"</i> - eventData[0].component.name <i>"Connector"</i> - eventData[0].component.evse.id <Configured evseld> - eventData[0].component.evse.connectorId <Configured connectorId> - eventData[0].variable.name <i>"AvailabilityState"</i> |
| Post scenario validations: - A message to report the state of the connector has been received. |

TC_G_09_CS: Change Availability EVSE - Operative to operative

| | |
|-------------------|--|
| Test case name | Change Availability EVSE - Operative to operative |
| Test case Id | TC_G_09_CS |
| Use case Id(s) | G03 |
| Requirement(s) | G03.FR.01, G03.FR.02, G03.FR.03, G03.FR.04 |
| System under test | Charging Station |
| Description | This test case covers how the CSMS requests the Charging Station to change the availability of one of the EVSEs from Operative to Operative. An EVSE is considered Operative in any status other than Faulted and Unavailable. |
| Purpose | To verify if the Charging Station is able to perform the change availability from Operative to Operative according to the mechanism as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a ChangeAvailabilityResponse | 1. The Test System sends a ChangeAvailabilityRequest with operationalStatus <i>Operative</i> and evse.id <i><Configured evseld></i> |

| |
|---|
| Tool validations |
| * Step 2: Message ChangeAvailabilityResponse - status <i>Accepted</i> |
| Post scenario validations: N/a |

TC_G_10_CS: Change Availability EVSE - Inoperative to inoperative

| | |
|-------------------|---|
| Test case name | Change Availability EVSE - Inoperative to inoperative |
| Test case Id | TC_G_10_CS |
| Use case Id(s) | G03 |
| Requirement(s) | G03.FR.01, G03.FR.02, G03.FR.03, G03.FR.04 |
| System under test | Charging Station |
| Description | This test case covers how the CSMS requests the Charging Station to change the availability of one of the EVSEs from Inoperative to Inoperative. An EVSE is considered Inoperative in status Faulted and Unavailable. |
| Purpose | To verify if the Charging Station is able to perform the change availability from inoperative to inoperative according to the mechanism as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: State is <i>Unavailable</i> for <Configured evseId> |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a ChangeAvailabilityResponse | 1. The Test System sends a ChangeAvailabilityRequest with operationalStatus <i>Inoperative</i> and evse.id <Configured evseId> |

| |
|--|
| Tool validations |
| * Step 2: Message ChangeAvailabilityResponse - status <i>Accepted</i> |
| Post scenario validations: - A message to report the state of a connector has been received for all connectors. |

TC_G_11_CS: Change Availability EVSE - With ongoing transaction

| | |
|-------------------|--|
| Test case name | Change Availability EVSE - With ongoing transaction |
| Test case Id | TC_G_11_CS |
| Use case Id(s) | G03 |
| Requirement(s) | G03.FR.01, G03.FR.02, G03.FR.04, G03.FR.05 |
| System under test | Charging Station |
| Description | This test case covers how the CSMS requests the Charging Station to change the availability of one of the EVSEs from Operative to Inoperative. An EVSE is considered Operative in any status other than Faulted and Unavailable. |
| Purpose | To verify if the Charging Station is able to perform the change availability during a transaction according to the mechanism as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: State is <i>EnergyTransferStarted</i> |

| Main (Test scenario) | |
|--|--|
| Charging Station | CSMS |
| 2. The Charging Station responds with a ChangeAvailabilityResponse | 1. The Test System sends a ChangeAvailabilityRequest with operationalStatus <i>Inoperative</i> and evse.id <i><Configured evseld></i> |
| <u>Note(s)</u> : Wait for <i><Configured Transaction Duration></i> | |
| 3. Execute Reusable State <i>StopAuthorized</i> | |
| 4. The Charging Station notifies the CSMS about the current state of the connectors of the configured evse. | 5. The Test System responds accordingly. |
| 6. Execute Reusable State <i>EVConnectedPostSession</i> | |
| 7. The Charging Station notifies the CSMS about the current state of the connectors of the configured evse. | 8. The Test System responds accordingly. |
| 9. Execute Reusable State <i>EVDisconnected</i> | |
| 10. The Charging Station notifies the CSMS about the current state of the connectors of the configured evse. | 11. The Test System responds accordingly. |
| 12. Execute Reusable State <i>ParkingBayUnoccupied</i> | |
| 13. The Charging Station notifies the CSMS about the current state of the connectors of the configured evse. | 14. The Test System responds accordingly. |
| <u>Note(s)</u> : Steps 4, 5, 7, 8, 10, 11, 13, and 14 will only be executed if the previous step ended the transaction | |

Tool validations

* Step 2:

Message **ChangeAvailabilityResponse**

- **status** *Scheduled*

* Step 4, 7, 10, 13:

Message: **StatusNotificationRequest**

- **connectorStatus** *Unavailable*

- **evseld** *<Configured evseld>*

- **connectorId** *<Configured connectorId>*

- when applicable this is also sent for other connectors of the same EVSE

Message: **NotifyEventRequest**

- **eventData[0].trigger** *Delta*

- **eventData[0].actualValue** *"Unavailable"*

- **eventData[0].component.name** *"Connector"*

- **eventData[0].variable.name** *"AvailabilityState"*

- **eventData[0].component.evse.id** *<Configured evseld>*

- **eventData[0].component.evse.connectorId** *<Configured connectorId>*

- when applicable this is also sent for other connectors of the same EVSE

Post scenario validations:

- A message to report the state of a connector has been received for all connectors.

TC_G_12_CS: Change Availability Charging Station - Operative to operative

| | |
|-------------------|--|
| Test case name | Change Availability Charging Station - Operative to operative |
| Test case Id | TC_G_12_CS |
| Use case Id(s) | G04 |
| Requirement(s) | G04.FR.01, G04.FR.02, G04.FR.03, G04.FR.04, G04.FR.05 |
| System under test | Charging Station |
| Description | <p>This test case describes how the CSMS requests the Charging Station to change the availability from inoperative to operative.</p> <p>A Charging Station is considered Operative when it is charging or ready for charging.</p> <p>A Charging Station is considered Inoperative when it does not allow any charging.</p> |
| Purpose | To verify if the Charging Station is able to perform the change availability from operative to operative according to the mechanism as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a ChangeAvailabilityResponse | 1. The Test System sends a ChangeAvailabilityRequest with operationalStatus <i>Operative</i> |

| |
|---|
| Tool validations |
| * Step 2: Message ChangeAvailabilityResponse - status <i>Accepted</i> |
| Post scenario validations: - A message to report the state of a connector has been received for all connectors. |

TC_G_13_CS: Change Availability Charging Station - Inoperative to inoperative

| | |
|-------------------|--|
| Test case name | Change Availability Charging Station - Inoperative to inoperative |
| Test case Id | TC_G_13_CS |
| Use case Id(s) | G04 |
| Requirement(s) | G04.FR.01, G04.FR.02, G04.FR.03, G04.FR.04, G04.FR.05 |
| System under test | Charging Station |
| Description | <p>This test case describes how the CSMS requests the Charging Station to change the availability from operative to inoperative.</p> <p>A Charging Station is considered Operative when it is charging or ready for charging.</p> <p>A Charging Station is considered Inoperative when it does not allow any charging.</p> |
| Purpose | To verify if the Charging Station is able to perform the change availability from Inoperative to Inoperative according to the mechanism as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: State is <i>Unavailable</i> |

| Main (Test scenario) | |
|--|--|
| Charging Station | CSMS |
| 2. The Charging Station responds with a ChangeAvailabilityResponse | 1. The Test System sends a ChangeAvailabilityRequest with operationalStatus <i>Inoperative</i> |
| 3. The Charging Station notifies the CSMS about the current state of all connectors. | 4. The Test System responds accordingly. |

| |
|--|
| Tool validations |
| <p>* Step 2: Message ChangeAvailabilityResponse - status <i>Accepted</i></p> <p>* Step 3: Message: StatusNotificationRequest - connectorStatus <i>Unavailable</i></p> <p>Message: NotifyEventRequest - eventData[0].trigger <i>Delta</i> - eventData[0].actualValue <i>"Unavailable"</i> - eventData[0].component.name <i>"ChargingStation"</i> - eventData[0].variable.name <i>"AvailabilityState"</i></p> |
| Post scenario validations: - A message to report the state of a connector has been received for all connectors. |

TC_G_14_CS: Change Availability Charging Station - With ongoing transaction

| | |
|-------------------|--|
| Test case name | Change Availability Charging Station - With ongoing transaction |
| Test case Id | TC_G_14_CS |
| Use case Id(s) | G04 |
| Requirement(s) | G04.FR.01, G04.FR.02, G04.FR.03, G04.FR.05, G04.FR.06 |
| System under test | Charging Station |
| Description | This test case covers how the CSMS requests the Charging Station to change the availability of one of the EVSEs from Operative to Inoperative. An EVSE is considered Operative in any status other than Faulted and Unavailable. |
| Purpose | To verify if the Charging Station is able to perform the change availability during a transaction according to the mechanism as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: State is <i>EnergyTransferStarted</i> |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a ChangeAvailabilityResponse | 1. The Test System sends a ChangeAvailabilityRequest with operationalStatus <i>Inoperative</i> |
| 3. The Charging Station notifies the CSMS about the current state of the connectors of the EVSEs that do not have an active transaction. | 4. The Test System responds accordingly. |
| <u>Note(s)</u> : Wait for <Configured Transaction Duration> | |
| 5. Execute Reusable State <i>StopAuthorized</i> | |
| 6. The Charging Station notifies the CSMS about the current state of the connectors of the configured evse. | 7. The Test System responds accordingly. |
| 8. Execute Reusable State <i>EVConnectedPostSession</i> | |
| 9. The Charging Station notifies the CSMS about the current state of the connectors of the configured evse. | 10. The Test System responds accordingly. |
| 11. Execute Reusable State <i>EVDisconnected</i> | |
| 12. The Charging Station notifies the CSMS about the current state of the connectors of the configured evse. | 13. The Test System responds accordingly. |
| 14. Execute Reusable State <i>ParkingBayUnoccupied</i> | |
| 15. The Charging Station notifies the CSMS about the current state of the connectors of the configured evse. | 16. The Test System responds accordingly. |
| <u>Note(s)</u> : Steps 6, 7, 9, 10, 12, 13, 15, and 16 will only be executed if the previous step ended the transaction | |

| Tool validations |
|--|
| <p>* Step 2:</p> <p>Message ChangeAvailabilityResponse</p> <ul style="list-style-type: none">- status <i>Scheduled</i> <p>* Step 7, 10, 13, 16:</p> <p>Message: StatusNotificationRequest</p> <ul style="list-style-type: none">- connectorStatus <i>Unavailable</i>- evseld not <i>0</i>- connectorId not <i>0</i> <p>Message: NotifyEventRequest</p> <ul style="list-style-type: none">- eventData[0].trigger <i>Delta</i>- eventData[0].actualValue <i>"Unavailable"</i>- eventData[0].component.name <i>"Connector"</i>- eventData[0].variable.name <i>"AvailabilityState"</i> |
| <p>Post scenario validations:</p> <ul style="list-style-type: none">- A message to report the state of a connector has been received for all connectors. |

TC_G_15_CS: Change Availability Connector - Operative to operative

| | |
|-------------------|--|
| Test case name | Change Availability Connector - Operative to operative |
| Test case Id | TC_G_15_CS |
| Use case Id(s) | G03 |
| Requirement(s) | G03.FR.01, G03.FR.02, G03.FR.03, G03.FR.04 |
| System under test | Charging Station |
| Description | This test case covers how the CSMS requests the Charging Station to change the availability of one of the EVSEs from Operative to Inoperative. An EVSE is considered Operative in any status other than Faulted and Unavailable. |
| Purpose | To verify if the Charging Station is able to perform the change availability from Operative to Operative of one connector according to the mechanism as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a ChangeAvailabilityResponse | 1. The Test System sends a ChangeAvailabilityRequest with operationalStatus <i>Operative</i> and evse.id <Configured evseId> and evse.connectorId <Configured connectorId> |

| |
|--|
| Tool validations |
| * Step 2: Message ChangeAvailabilityResponse - status <i>Accepted</i> |
| Post scenario validations: - A message to report the state of a connector has been received for all connectors. |

TC_G_16_CS: Change Availability Connector - Inoperative to inoperative

| | |
|-------------------|--|
| Test case name | Change Availability Connector - Inoperative to inoperative |
| Test case Id | TC_G_16_CS |
| Use case Id(s) | G03 |
| Requirement(s) | G03.FR.01, G03.FR.02, G03.FR.03, G03.FR.04 |
| System under test | Charging Station |
| Description | This test case covers how the CSMS requests the Charging Station to change the availability of one of the EVSEs from Operative to Inoperative. An EVSE is considered Operative in any status other than Faulted and Unavailable. |
| Purpose | To verify if the Charging Station is able to perform the change availability from inoperative to inoperative on one connector according to the mechanism as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: State is <i>Unavailable</i> |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a ChangeAvailabilityResponse | 1. The Test System sends a ChangeAvailabilityRequest with operationalStatus <i>Inoperative</i> and evse.id <i><Configured evseId></i> and evse.connectorId <i><Configured connectorId></i> |

| |
|--|
| Tool validations |
| * Step 2: Message ChangeAvailabilityResponse - status <i>Accepted</i> |
| Post scenario validations: - A message to report the state of a connector has been received for all connectors. |

TC_G_17_CS: Change Availability Connector - With ongoing transaction

| | |
|-------------------|--|
| Test case name | Change Availability Connector - With ongoing transaction |
| Test case Id | TC_G_17_CS |
| Use case Id(s) | G03 |
| Requirement(s) | G03.FR.01, G03.FR.02, G03.FR.04, G03.FR.05 |
| System under test | Charging Station |
| Description | This test case covers how the CSMS requests the Charging Station to change the availability of one of the EVSEs from Operative to Inoperative. An EVSE is considered Operative in any status other than Faulted and Unavailable. |
| Purpose | To verify if the Charging Station is able to perform the change availability during a transaction according to the mechanism as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: State is <i>EnergyTransferStarted</i> |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a ChangeAvailabilityResponse | 1. The Test System sends a ChangeAvailabilityRequest with operationalStatus <i>Inoperative</i> and evse.id <i><Configured evseld></i> and evse.connectorId <i><Configured connectorId></i> |
| <u>Note(s)</u> : Wait for <i><Configured Transaction Duration></i> | |
| 3. Execute Reusable State <i>StopAuthorized</i> | |
| 4. The Charging Station notifies the CSMS about the current state of the connectors of the configured evse. | 5. The Test System responds accordingly. |
| 6. Execute Reusable State <i>EVConnectedPostSession</i> | |
| 7. The Charging Station notifies the CSMS about the current state of the connectors of the configured evse. | 8. The Test System responds accordingly. |
| 9. Execute Reusable State <i>EVDisconnected</i> | |
| 10. The Charging Station notifies the CSMS about the current state of the connectors of the configured evse. | 11. The Test System responds accordingly. |
| 12. Execute Reusable State <i>ParkingBayUnoccupied</i> | |
| 13. The Charging Station notifies the CSMS about the current state of the connectors of the configured evse. | 14. The Test System responds accordingly. |
| <u>Note(s)</u> : Steps 4, 5, 7, 8, 10, 11, 13, and 14 will only be executed if the previous step ended the transaction | |

| Tool validations |
|--|
| <p>* Step 2:</p> <p>Message ChangeAvailabilityResponse</p> <ul style="list-style-type: none">- status <i>Scheduled</i> <p>* Step 7:</p> <p>Message: StatusNotificationRequest</p> <ul style="list-style-type: none">- connectorStatus <i>Unavailable</i>- evseld <i><Configured evseld></i>- connectorId <i><Configured connectorId></i>- when applicable this is also sent for other connectors of the same EVSE <p>Message: NotifyEventRequest</p> <ul style="list-style-type: none">- eventData[0].trigger <i>Delta</i>- eventData[0].actualValue <i>"Unavailable"</i>- eventData[0].component.name <i>"Connector"</i>- eventData[0].component.evse <i>not omit</i>- eventData[0].component.evse.id <i><Configured evseld></i>- eventData[0].component.evse.connectorId <i><Configured connectorId></i>- eventData[0].variable.name <i>"AvailabilityState"</i>- when applicable this is also sent for other connectors of the same EVSE |
| <p>Post scenario validations:</p> <ul style="list-style-type: none">- A message to report the state of a connector has been received for all connectors. |

TC_G_18_CS: Change Availability EVSE - state persists across reboot

| | |
|-------------------|--|
| Test case name | Change Availability EVSE - state persists across reboot |
| Test case Id | TC_G_18_CS |
| Use case Id(s) | G03 |
| Requirement(s) | G03.FR.08. G01.FR.01 |
| System under test | Charging Station |
| Description | This test case covers how the CSMS requests the Charging Station to change the availability of one of the EVSEs from Operative to Inoperative. An EVSE is considered Operative in any status other than Faulted and Unavailable. |
| Purpose | To verify if the Charging Station sets the availability persistent across reboot/power loss as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: state is <i>Unavailable</i> |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a ChangeAvailabilityResponse | 1. The Test System sends a ChangeAvailabilityRequest with operationalStatus <i>Inoperative</i> AND evse.id <Configured evseld> |
| 3. The Charging Station notifies the CSMS about the current state of all connectors. | 4. The Test System responds accordingly. |
| 5. Execute Reusable State <i>Booted</i> <u>Note(s):</u> - After booting the charging station should send the following status: Message: StatusNotificationRequest - connectorStatus <i>Unavailable</i> - evseld <Configured evseld> Message: NotifyEventRequest - eventData[0].trigger <i>Delta</i> - eventData[0].actualValue "Unavailable" - eventData[0].component.name "Connector" - eventData[0].component.evse.id <Configured evseld> - eventData[0].variable.name "AvailabilityState" | |

| Tool validations |
|--|
| <p>* Step 2:</p> <p>Message ChangeAvailabilityResponse</p> <ul style="list-style-type: none">- status <i>Accepted</i> <p>* Step 3:</p> <p>Message: StatusNotificationRequest</p> <ul style="list-style-type: none">- evseld not <i>0</i>- connectorId not <i>0</i>- connectorStatus <i>Unavailable</i> for evseld <i><Configured evseld></i>- connectorStatus <i>Available</i> for evseld not <i><Configured evseld></i> <p>Message: NotifyEventRequest</p> <ul style="list-style-type: none">- eventData[0].actualValue <i>Unavailable</i> for evseld <i><Configured evseld></i>- eventData[0].actualValue <i>Available</i> for evseld not <i><Configured evseld></i> |
| <p>Post scenario validations:</p> <ul style="list-style-type: none">- A message to report the state of a connector has been received for all connectors. |

TC_G_19_CS: Change Availability Connector - state persists across reboot

| | |
|-------------------|---|
| Test case name | Change Availability Connector - state persists across reboot |
| Test case Id | TC_G_19_CS |
| Use case Id(s) | G03 |
| Requirement(s) | G03.FR.08 |
| System under test | Charging Station |
| Description | This test case covers how the CSMS requests the Charging Station to change the availability of one of the Connectors of one EVSE from Operative to Inoperative. A Connector is considered Operative in any status other than Faulted and Unavailable. |
| Purpose | To verify if the Charging Station sets the availability persistent across reboot/power loss as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: state is <i>Unavailable</i> |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>Booting</i> | |
| 2. The Charging Station sends a BootNotificationRequest | 3. The Test System responds with a BootNotificationResponse . |
| 4. The Charging Station reports the status of all its connectors. | 5. The Test System responds accordingly. |
| 6. The Charging Station sends a SecurityEventNotificationRequest | 7. The Test System responds with a SecurityEventNotificationResponse |

| |
|---|
| Tool validations |
| <p>* Step 4:</p> <p>Message: StatusNotificationRequest</p> <ul style="list-style-type: none"> - evseld not 0 - connectorId not 0 - connectorStatus <i>Unavailable</i> for evseld <Configured evseld> and for connectorId <Configured ConnectorId> - connectorStatus <i>Available</i> for evseld not <Configured evseld> and for connectorId <Configured ConnectorId> <p>Message: NotifyEventRequest</p> <ul style="list-style-type: none"> - eventData[0].actualValue <i>Unavailable</i> for evseld <Configured evseld> and for connectorId <Configured ConnectorId> - eventData[0].actualValue <i>Available</i> for evseld not <Configured evseld> and for connectorId <Configured ConnectorId> <p>* Step 6:</p> <p>Message: SecurityEventNotificationRequest</p> <ul style="list-style-type: none"> - type "StartupOfTheDevice" or type "ResetOrReboot" |
| <p>Post scenario validations:</p> <ul style="list-style-type: none"> - A message to report the state of a connector has been received for all connectors. |

TC_G_20_CS: Connector status Notification - Lock Failure

| | |
|-------------------|---|
| Test case name | Connector status Notification - Lock Failure |
| Test case Id | TC_G_20_CS |
| Use case Id(s) | G05 |
| Requirement(s) | G05.FR.01, G05.FR.02 |
| System under test | Charging Station |
| Description | This test case describes how the EV Driver is prevented from starting a charge session at the Charging Station while the Connector is not locked properly. |
| Purpose | To verify if the Charging Station does not start charging and notifies the CSMS when a connector is not locked properly as described at the OCPP specification. |
| Prerequisite(s) | <ul style="list-style-type: none"> - Charging Station has the ConnectorPlugRetentionLock component defined in its Device Model. - MonitoringLevel is set to a level that a connector lock event failure will be reported. |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: State is <i>EVConnectedPreSession</i> <u>Note(s):</u> - Cable should not be fully plugged in so it cannot lock properly |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>Authorized</i> | |
| 2. The Charging Station sends a NotifyEventRequest | 3 . The Test System responds with a NotifyEventResponse |

| |
|--|
| Tool validations |
| * Step 2: Message: NotifyEventRequest - eventData[0].trigger <i>Delta</i> - eventData[0].component.name <i>"ConnectorPlugRetentionLock"</i> - eventData[0].variable.name <i>"Problem"</i> - eventData[0].actualValue <i>"true"</i> |
| Post scenario validations: - The charging station did not start charging |

TC_G_21_CS: Change Availability Charging Station - state persists across reboot

| | |
|-------------------|--|
| Test case name | Change Availability Charging Station - state persists across reboot |
| Test case Id | TC_G_21_CS |
| Use case Id(s) | G04 |
| Requirement(s) | G04.FR.09 |
| System under test | Charging Station |
| Description | This test case covers how the CSMS requests the Charging Station to change the availability from Operative to Inoperative. An EVSE is considered Operative in any status other than Faulted and Unavailable. |
| Purpose | To verify if the Charging Station sets the availability persistent across reboot/power loss as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: State is <i>Unavailable</i> |

| Main (Test scenario) | |
|---|---|
| Charging Station | CSMS |
| 1. Execute Reusable State <i>Booting</i> | |
| 2. The Charging Station sends a BootNotificationRequest . | 3. The Test System responds with a BootNotificationResponse . |
| 4. The Charging Station reports the status of all its connectors. | 5. The Test System responds accordingly. |
| 6. The Charging Station sends a SecurityEventNotificationRequest | 7. The Test System responds with a SecurityEventNotificationResponse |

| |
|--|
| Tool validations |
| <p>* Step 4:</p> <p>Message: StatusNotificationRequest</p> <ul style="list-style-type: none"> - evseld not 0 - connectorId not 0 - connectorStatus <i>Unavailable</i> <p>Message: NotifyEventRequest</p> <ul style="list-style-type: none"> - eventData[0].trigger <i>Delta</i> - eventData[0].actualValue <i>"Unavailable"</i> - eventData[0].variable.name <i>"AvailabilityState"</i> <p>* Step 6:</p> <p>Message: SecurityEventNotificationRequest</p> <ul style="list-style-type: none"> - type <i>"StartupOfTheDevice"</i> or type <i>"ResetOrReboot"</i> |
| <p>Post scenario validations:</p> <ul style="list-style-type: none"> - A message to report the state of a connector has been received for all connectors. |

H Reservation

TC_H_01_CS: Reserve a specific EVSE - Accepted - Valid idToken

| | |
|-------------------|---|
| Test case name | Reserve a specific EVSE - Accepted - Valid idToken |
| Test case Id | TC_H_01_CS |
| Use case Id(s) | H01(S2), H03 |
| Requirement(s) | H01.FR.15, H03.FR.01, H03.FR.09, H03.FR.10 |
| System under test | Charging Station |
| Description | The CSMS is able to reserve a specific EVSE for a specific IdToken by sending a ReserveNowRequest containing an evseld. |
| Purpose | To verify if the Charging Station is able to reserve a specific EVSE, until the EV Driver with the specified IdToken arrives. |
| Prerequisite(s) | The configuration variable ReservationCtrlr.ReservationAvailable is implemented with value <i>true</i> |

| |
|---|
| Before (Preparations) |
| Configuration State: ReservationCtrlr.ReservationEnabled is <i>true</i> (If implemented) |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>Reserved</i> for <Configured evseld> | |
| 2. Execute Reusable State <i>Authorized</i> | |
| <u>Note(s)</u> : - <Configured valid idToken fields> are used for the authorization. | |
| 3. Execute Reusable State <i>EnergyTransferStarted</i> | |

| |
|--|
| Tool validations |
| <p>* Step 2:</p> <p>After authorization, connector status must change from Reserved to Available</p> <p>Message: StatusNotificationRequest</p> <ul style="list-style-type: none"> - evseld <configured evseld> - connectorId <configured connectorId> - connectorStatus must be Available <p>Message: NotifyEventRequest</p> <ul style="list-style-type: none"> - eventData[0].trigger must be Delta - eventData[0].actualValue must be Available - eventData[0].component.name must be Connector - eventData[0].variable.name must be AvailabilityState - evse.id <configured evseld> - connector.id <configured connectorId> <p>Post scenario validations: N/a</p> |

TC_H_02_CS: Reserve a specific EVSE - Accepted - Different idToken

| | |
|-------------------|--|
| Test case name | Reserve a specific EVSE - Accepted - Different idToken |
| Test case Id | TC_H_02_CS |
| Use case Id(s) | H01(S2), H03 |
| Requirement(s) | H03.FR.01, F01.FR.22 |
| System under test | Charging Station |
| Description | <p>The CSMS is able to reserve a specific EVSE for a specific IdToken by sending a ReserveNowRequest containing an evseld.</p> <p>Starting a transaction can be done in two ways (this is configurable by the Test System;</p> <p>A. Using local authorization</p> <p>B. Using a RequestStartTransactionRequest</p> |
| Purpose | To verify if the Charging Station rejects all idToken, except the one specified for the reserved EVSE. EV is plugged in before authorization to check that station is able to handle this correctly. When TxStartPoint contains EVConnected this triggers starting of a transaction, but charging must not be allowed when idToken does not match reservation. |
| Prerequisite(s) | The configuration variable ReservationCtrlr.ReservationAvailable is implemented with value <i>true</i> |

Before (Preparations)

| |
|---|
| Configuration State: ReservationCtrlr.ReservationEnabled is <i>true</i> (If implemented) |
| Memory State: N/a |
| Reusable State(s): State is <Configured evseld> is <i>Reserved</i> for <Configured valid_idtoken1_idtoken> State is <i>EVConnectedPreSession</i> |

Main A (Test scenario)

| | |
|--|------|
| Charging Station | CSMS |
| <u>Manual action:</u> Authorize with <Configured valid_idtoken2_idtoken>. | |
| Execute reusable state <i>Authorized</i> | |
| <u>Note(s):</u> The test is a PASS, if the Test System does not receive an a TransactionEventRequest with chargingState = Charging within the configured messageTimeout. | |

Tool validations

N/a

Main B (Test scenario)

| | |
|---|--|
| Charging Station | CSMS |
| 2. The Charging Station responds with a RequestStartTransactionResponse | 1. The Test System sends a RequestStartTransactionRequest with idToken.idToken <Configured valid_idtoken2_idtoken> idToken.type <Configured valid_idtoken2_type> evseld <Configured evseld> |

Tool validations

* Step 2:
Message: **RequestStartTransactionResponse**
- **status** must be *Rejected*

Post scenario validations:
N/a

TC_H_03_CS: Reserve a specific EVSE - Occupied - EVSE Reserved

| | |
|-------------------|---|
| Test case name | Reserve a specific EVSE - Occupied - EVSE Reserved |
| Test case Id | TC_H_03_CS |
| Use case Id(s) | H01(S2) |
| Requirement(s) | H01.FR.11 |
| System under test | Charging Station |
| Description | The CSMS is able to reserve a specific EVSE for a specific IdToken by sending a ReserveNowRequest containing an evseld. |
| Purpose | To verify if the Charging Station is able to respond with status Occupied, when the requested EVSE is already reserved. |
| Prerequisite(s) | The configuration variable ReservationCtrlr.ReservationAvailable is implemented with value <i>true</i> |

| |
|---|
| Before (Preparations) |
| Configuration State: ReservationCtrlr.ReservationEnabled is <i>true</i> (If implemented) |
| Memory State: <Configured evseld> is <i>Reserved</i> |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a ReserveNowResponse | 1. The Test System sends a ReserveNowRequest with evseld is <Configured evseld> idToken.idToken <Configured valid_idtoken2_idtoken> idToken.type <Configured valid_idtoken2_type> |

| |
|--|
| Tool validations |
| * Step 2: Message: ReserveNowResponse - status must be <i>Occupied</i> |
| Post scenario validations: N/a |

TC_H_04_CS: Reserve a specific EVSE - Occupied - EVSE Occupied

| | |
|-------------------|---|
| Test case name | Reserve a specific EVSE - Occupied - EVSE Occupied |
| Test case Id | TC_H_04_CS |
| Use case Id(s) | H01(S2) |
| Requirement(s) | H01.FR.13 |
| System under test | Charging Station |
| Description | The CSMS is able to reserve a specific EVSE for a specific IdToken by sending a ReserveNowRequest containing an evseld. |
| Purpose | To verify if the Charging Station is able to respond with status Occupied, when the requested EVSE is occupied. |
| Prerequisite(s) | The configuration variable ReservationCtrlr.ReservationAvailable is implemented with value <i>true</i> |

| |
|---|
| Before (Preparations) |
| Configuration State: ReservationCtrlr.ReservationEnabled is <i>true</i> (If implemented) |
| Memory State: State is <i>EnergyTransferStarted</i> |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|---|---|
| Charging Station | CSMS |
| 2. The Charging Station responds with a ReserveNowResponse | 1. The Test System sends a ReserveNowRequest with evseld is <Configured evseld> idToken.idToken <Configured valid_idtoken2_idtoken> idToken.type <Configured valid_idtoken2_type> |

| |
|--|
| Tool validations |
| * Step 2: Message: ReserveNowResponse - status must be <i>Occupied</i> |
| Post scenario validations: N/a |

TC_H_06_CS: Reserve a specific EVSE - Unavailable

| | |
|-------------------|---|
| Test case name | Reserve a specific EVSE - Unavailable |
| Test case Id | TC_H_06_CS |
| Use case Id(s) | H01(S2) |
| Requirement(s) | H01.FR.14 |
| System under test | Charging Station |
| Description | The CSMS is able to reserve a specific EVSE for a specific IdToken by sending a ReserveNowRequest containing an evseld. |
| Purpose | To verify if the Charging Station is able to respond with status Unavailable, when the requested EVSE is unavailable. |
| Prerequisite(s) | The configuration variable ReservationCtrlr.ReservationAvailable is implemented with value <i>true</i> |

| |
|---|
| Before (Preparations) |
| Configuration State: ReservationCtrlr.ReservationEnabled is <i>true</i> (If implemented) |
| Memory State: <Configured evseld> is <i>Unavailable</i> |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a ReserveNowResponse | 1. The Test System sends a ReserveNowRequest with evseld is <Configured evseld> idToken.idToken <Configured valid_idtoken2_idtoken> idToken.type <Configured valid_idtoken2_type> |

| |
|---|
| Tool validations |
| * Step 2: Message: ReserveNowResponse - status must be <i>Unavailable</i> |
| Post scenario validations: N/a |

TC_H_07_CS: Reserve a specific EVSE - Reservation Ended / not used

| | |
|-------------------|---|
| Test case name | Reserve a specific EVSE - Reservation Ended / not used |
| Test case Id | TC_H_07_CS |
| Use case Id(s) | H01(S2), H04 |
| Requirement(s) | H04.FR.01,H04.FR.02,H04.FR.03 |
| System under test | Charging Station |
| Description | The CSMS is able to reserve a specific EVSE for a specific IdToken by sending a ReserveNowRequest containing an evseld. |
| Purpose | To verify if the Charging Station is able to end the reservation, when the EV Driver with the specified IdToken arrives, does not arrive before the set expiryDateTime is reached. |
| Prerequisite(s) | The configuration variable ReservationCtrlr.ReservationAvailable is implemented with value <i>true</i> |

| |
|---|
| Before (Preparations) |
| Configuration State: ReservationCtrlr.ReservationEnabled is <i>true</i> (If implemented) |
| Memory State: <Configured evseld> is <i>Reserved</i> |
| Reusable State(s): N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Charging Station notifies the CSMS about the status change of the connector. <u>Note(s):</u> - The Test System expects that the Charging Station sets the <i>availabilityState</i> of the EVSE and corresponding connectors back to <i>Available</i> after the expiry time of 60 seconds is reached. | 2. The Test System responds accordingly. |
| 3. The Charging Station sends a ReservationStatusUpdateRequest . | 4. The Test System responds with a ReservationStatusUpdateResponse . |
| 5. Execute Reusable State <i>Authorized</i> <u>Note(s):</u> - <Configured valid idtoken fields2> are used for the authorization. | |
| 6. Execute Reusable State <i>EnergyTransferStarted</i> | |

| Tool validations |
|--|
| <p>* Step 1:</p> <p>Message: StatusNotificationRequest</p> <ul style="list-style-type: none">- connectorStatus must be <i>Available</i> <p>Message: NotifyEventRequest</p> <ul style="list-style-type: none">- eventData[0].trigger must be <i>Delta</i>- eventData[0].actualValue must be <i>Available</i>- eventData[0].component.name must be <i>Connector</i>- eventData[0].variable.name must be <i>AvailabilityState</i> <p>(Reporting the AvailabilityState of the EVSE component itself is optional.)</p> <p>Message: NotifyEventRequest</p> <ul style="list-style-type: none">- eventData[0].trigger must be <i>Delta</i>- eventData[0].actualValue must be <i>Available</i>- eventData[0].component.name must be <i>EVSE</i>- eventData[0].variable.name must be <i>AvailabilityState</i> <p>* Step 3:</p> <p>Message: ReservationStatusUpdateRequest</p> <ul style="list-style-type: none">- reservationId must be <i><Generated reservationId></i>- reservationUpdateStatus must be <i>Expired</i> |
| <p>Post scenario validations:</p> <p>N/a</p> |

TC_H_08_CS: Reserve an unspecified EVSE - Accepted

| | |
|-------------------|--|
| Test case name | Reserve an unspecified EVSE - Accepted |
| Test case Id | TC_H_08_CS |
| Use case Id(s) | H01(S1), H03 |
| Requirement(s) | H01.FR.04,H01.FR.07,H01.FR.15,H03.FR.03 |
| System under test | Charging Station |
| Description | The CSMS is able to reserve an unspecified EVSE for a specific IdToken by sending a ReserveNowRequest without an evseld. |
| Purpose | To verify if the Charging Station is able to reserve an unspecified EVSE, until the EV Driver with the specified IdToken arrives. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The configuration variable ReservationCtrlr.ReservationAvailable is implemented with value <i>true</i> - The Charging Station has the configuration variable ReservationNonEvseSpecific implemented with value <i>true</i> |

| |
|---|
| Before (Preparations) |
| Configuration State: ReservationCtrlr.ReservationEnabled is <i>true</i> (If implemented) |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a ReserveNowResponse | 1. The Test System sends a ReserveNowRequest with evseld is Omitted idToken.idToken <Configured valid_idtoken_idtoken> idToken.type <Configured valid_idtoken_type> |
| 3. The Charging Station notifies the CSMS about the status change of the connector. <u>Note(s):</u> - If the Charging Station has only one EVSE, it sets the <i>availabilityState</i> of the EVSE and corresponding connectors to <i>Reserved</i> . - Reporting the <i>AvailabilityState</i> of the EVSE component itself is optional. | 4. The Test System responds accordingly. |
| 5. Execute Reusable State <i>Authorized</i> <u>Note(s):</u> - <Configured valid_idtoken_idtoken> is used for the authorization. | |
| 6. Execute Reusable State <i>EnergyTransferStarted</i> | |

| Tool validations |
|--|
| <div>* Step 2: Message: ReserveNowResponse - status must be <i>Accepted</i></div> <div>* Step 3: (Optional) Message: StatusNotificationRequest - connectorStatus must be <i>Reserved</i> - evseld must be <i><Configured evseld></i> - connectorId must be <i><Configured connectorId></i></div> <div>Message: NotifyEventRequest - trigger must be <i>Delta</i> - actualValue must be <i>"Reserved"</i> - component.name must be <i>"Connector"</i> - evse.id must be <i><Configured evseld></i> - eves.connectorId must be <i><Configured connectorId></i> - variable.name must be <i>"AvailabilityState"</i> (Optional) Message: NotifyEventRequest - eventData[0].trigger must be <i>Delta</i> - eventData[0].actualValue must be <i>Available</i> - eventData[0].component.name must be <i>EVSE</i> - eventData[0].variable.name must be <i>AvailabilityState</i></div> |
| <div>Post scenario validations: N/a</div> |

TC_H_09_CS: Reserve an unspecified EVSE - Occupied - EVSE Reserved

| | |
|-------------------|--|
| Test case name | Reserve an unspecified EVSE - Occupied - EVSE Reserved |
| Test case Id | TC_H_09_CS |
| Use case Id(s) | H01(S1) |
| Requirement(s) | H01.FR.11 |
| System under test | Charging Station |
| Description | The CSMS is able to reserve an unspecified EVSE for a specific IdToken by sending a ReserveNowRequest without an evseld. |
| Purpose | To verify if the Charging Station is able to respond with status Occupied, when all EVSE are already reserved. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The configuration variable ReservationCtrlr.ReservationAvailable is implemented with value <i>true</i> - The Charging Station has the configuration variable ReservationNonEvseSpecific implemented with value <i>true</i> |

Before (Preparations)

Configuration State:
ReservationCtrlr.ReservationEnabled is *true* (If implemented)

Memory State:
 N/a

Reusable State(s):
 All EVSE are *Reserved*

Main (Test scenario)

| Charging Station | CSMS |
|---|---|
| 2. The Charging Station responds with a ReserveNowResponse | 1. The Test System sends a ReserveNowRequest with evseld is Omitted idToken.idToken <Configured valid_idtoken_idtoken> idToken.type <Configured valid_idtoken_type> |

Tool validations

* Step 2:
 Message: **ReserveNowResponse**
 - **status** must be *Occupied*

Post scenario validations:
 N/a

TC_H_10_CS: Reserve an unspecified EVSE - Occupied - EVSE Occupied

| | |
|-------------------|--|
| Test case name | Reserve an unspecified EVSE - Occupied - EVSE Occupied |
| Test case Id | TC_H_10_CS |
| Use case Id(s) | H01(S1) |
| Requirement(s) | H01.FR.13 |
| System under test | Charging Station |
| Description | The CSMS is able to reserve an unspecified EVSE for a specific IdToken by sending a ReserveNowRequest without an evseld. |
| Purpose | To verify if the Charging Station is able to respond with status Occupied, when all EVSE are occupied. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The configuration variable ReservationCtrlr.ReservationAvailable is implemented with value <i>true</i> - The Charging Station has the configuration variable ReservationNonEvseSpecific implemented with value <i>true</i> |

Before (Preparations)

Configuration State:
ReservationCtrlr.ReservationEnabled is *true* (If implemented)

Memory State:
 N/a

Reusable State(s):
 State is *EnergyTransferStarted* for all EVSE

Main (Test scenario)

| Charging Station | CSMS |
|---|--|
| 2. The Charging Station responds with a ReserveNowResponse | 1. The Test System sends a ReserveNowRequest with evseld is Omitted idToken.idToken <Configured valid_idtoken_idtoken> idToken.type <Configured valid_idtoken_type> expiryDateTime <Configured expiryDateTime> |

Tool validations

* Step 2:
 Message: **ReserveNowResponse**
 - **status** must be *Occupied*

Post scenario validations:
 N/a

TC_H_12_CS: Reserve an unspecified EVSE - Unavailable

| | |
|-------------------|--|
| Test case name | Reserve an unspecified EVSE - Unavailable |
| Test case Id | TC_H_12_CS |
| Use case Id(s) | H01(S1) |
| Requirement(s) | H01.FR.14 |
| System under test | Charging Station |
| Description | The CSMS is able to reserve an unspecified EVSE for a specific IdToken by sending a ReserveNowRequest without an evseld. |
| Purpose | To verify if the Charging Station is able to respond with status Unavailable, when all EVSE are unavailable. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The configuration variable ReservationCtrlr.ReservationAvailable is implemented with value <i>true</i> - The Charging Station has the configuration variable ReservationNonEvseSpecific implemented with value <i>true</i> |

| |
|---|
| Before (Preparations) |
| Configuration State: ReservationCtrlr.ReservationEnabled is <i>true</i> (If implemented) |
| Memory State: Charging Station is <i>Unavailable</i> |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|---|---|
| Charging Station | CSMS |
| 2. The Charging Station responds with a ReserveNowResponse | 1. The Test System sends a ReserveNowRequest with evseld is Omitted idToken.idToken <Configured valid_idtoken_idtoken> idToken.type <Configured valid_idtoken_type> |

| |
|---|
| Tool validations |
| * Step 2: Message: ReserveNowResponse - status must be <i>Unavailable</i> |
| Post scenario validations: N/a |

TC_H_13_CS: Reserve an unspecified EVSE - Rejected

| | |
|-------------------|--|
| Test case name | Reserve an unspecified EVSE - Rejected |
| Test case Id | TC_H_13_CS |
| Use case Id(s) | H01(S1) |
| Requirement(s) | H01.FR.19 |
| System under test | Charging Station |
| Description | The CSMS is able to reserve an unspecified EVSE for a specific IdToken by sending a ReserveNowRequest without an evseld. |
| Purpose | To verify if the Charging Station is able to respond with status Rejected, when it does not support reserving an unspecified EVSE. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The configuration variable ReservationCtrlr.ReservationAvailable is implemented with value <i>true</i> - The Charging Station does NOT have the configuration variable ReservationNonEvseSpecific implemented OR the Charging Station does have it implemented with value <i>false</i> |

Before (Preparations)

Configuration State:
ReservationCtrlr.ReservationEnabled is *true* (If implemented)

Memory State:
 N/a

Reusable State(s):
 N/a

Main (Test scenario)

| Charging Station | CSMS |
|---|---|
| 2. The Charging Station responds with a ReserveNowResponse | 1. The Test System sends a ReserveNowRequest with evseld is Omitted idToken.idToken <Configured valid_idtoken_idtoken> idToken.type <Configured valid_idtoken_type> |

Tool validations

* Step 2:
 Message: **ReserveNowResponse**
 - **status** must be *Rejected*

Post scenario validations:
 N/a

TC_H_14_CS: Reserve an unspecified EVSE - Amount of EVSEs available equals the amount of reservations

| | |
|-------------------|---|
| Test case name | Reserve an unspecified EVSE - Amount of EVSEs available equals the amount of reservations |
| Test case Id | TC_H_14_CS |
| Use case Id(s) | H01(S1) |
| Requirement(s) | H01.FR.20 |
| System under test | Charging Station |
| Description | The CSMS is able to reserve an unspecified EVSE for a specific IdToken by sending a ReserveNowRequest without an evseld. |
| Purpose | To verify if the Charging Station is able to set all available EVSE to reserved, when the amount of EVSEs available equals the amount of reservations. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The configuration variable ReservationCtrlr.ReservationAvailable is implemented with value <i>true</i> - The Charging Station has the configuration variable ReservationNonEvseSpecific implemented with value <i>true</i> |

| |
|---|
| Before (Preparations) |
| Configuration State: ReservationCtrlr.ReservationEnabled is <i>true</i> (If implemented) |
| Memory State: N/a |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|--|---|
| Charging Station | CSMS |
| 2. The Charging Station responds with a ReserveNowResponse | <p>1. The Test System sends a ReserveNowRequest with evseld is Omitted idToken is <A different idToken for every reservation being made></p> <p><u>Note(s):</u> - This step will be executed the amount of times equal to the amount of EVSE the Charging Station has.</p> |
| 3. The Charging Station notifies the CSMS about the current state of all its connectors (and optionally also the state of all EVSE). | 4. The Test System responds accordingly. |

| |
|--|
| Tool validations |
| <p>* Step 2:</p> <p>Message: ReserveNowResponse</p> <ul style="list-style-type: none"> - status must be <i>Accepted</i> <p>* Step 3:</p> <p>Message: StatusNotificationRequest</p> <ul style="list-style-type: none"> - connectorStatus must be <i>Reserved</i> <p>Message: NotifyEventRequest</p> <ul style="list-style-type: none"> - eventData[0].trigger must be <i>Delta</i> - eventData[0].actualValue must be <i>Reserved</i> - eventData[0].component.name must be <i>Connector</i> - eventData[0].variable.name must be <i>AvailabilityState</i> <p>(Optional)</p> <p>Message: NotifyEventRequest</p> <ul style="list-style-type: none"> - eventData[0].trigger must be <i>Delta</i> - eventData[0].actualValue must be <i>Reserved</i> - eventData[0].component.name must be <i>EVSE</i> - eventData[0].variable.name must be <i>AvailabilityState</i> |

| |
|--|
| Tool validations |
| Post scenario validations: N/a |

TC_H_15_CS: Reserve a connector with a specific type - Success

| | |
|-------------------|--|
| Test case name | Reserve a connector with a specific type - Success |
| Test case Id | TC_H_15_CS |
| Use case Id(s) | H01(S3), H03 |
| Requirement(s) | H01.FR.06,H01.FR.09,H01.FR.15,H03.FR.02 |
| System under test | Charging Station |
| Description | The CSMS is able to reserve an EVSE with a connector with a specific type for a specific IdToken by sending a ReserveNowRequest with a connectorType. |
| Purpose | To verify if the Charging Station is able to reserve an EVSE with a connector with a specific type, until the EV Driver with the specified IdToken arrives. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The configuration variable ReservationCtrlr.ReservationAvailable is implemented with value <i>true</i> - The Charging Station supports the reservation of a specific connector type, that is part of the ConnectorEnumType. |

Before (Preparations)

Configuration State:

ReservationCtrlr.ReservationEnabled is *true* (If implemented)

Memory State:

N/a

Reusable State(s):

N/a

Main (Test scenario)

| Charging Station | CSMS |
|---|---|
| 2. The Charging Station responds with a ReserveNowResponse | 1. The Test System sends a ReserveNowRequest with connectorType is <Configured connectorType> idToken.idToken <Configured valid_idtoken_idtoken> idToken.type <Configured valid_idtoken_type> |
| 3. The Charging Station notifies the CSMS about the status change of the connector. <u>Note(s):</u> - If the Charging Station has only one available connector of the specified connectorType, it sets the availabilityState of the corresponding EVSE and all connectors of the specified type to Reserved. AND If the EVSE has more connector(s) with a different connectorType, the Charging Station must set these other connector(s) to Unavailable. - Reporting the AvailabilityState of the EVSE component itself is optional. | 4. The Test System responds accordingly. |
| 5. Execute Reusable State <i>Authorized</i> <u>Note(s):</u> - <Configured valid idToken fields> are used for the authorization. | |
| 6. Execute Reusable State <i>EnergyTransferStarted</i> | |

| Tool validations |
|--|
| <div>* Step 2: Message: ReserveNowResponse - status must be <i>Accepted</i></div> <div>* Step 3: Message: StatusNotificationRequest - connectorStatus must be <i>Reserved</i> - evseld must be <i><Configured evseld></i> - connectorId must be <i><Configured connectorId></i> Message: NotifyEventRequest - trigger must be <i>Delta</i> - actualValue must be <i>"Reserved"</i> - component.name must be <i>"Connector"</i> - evse.id must be <i><Configured evseld></i> - eves.connectorId must be <i><Configured connectorId></i> - variable.name must be <i>"AvailabilityState"</i> (Optional) Message: NotifyEventRequest - eventData[0].trigger must be <i>Delta</i> - eventData[0].actualValue must be <i>Available</i> - eventData[0].component.name must be <i>EVSE</i> - eventData[0].variable.name must be <i>AvailabilityState</i></div> |
| <div>Post scenario validations: N/a</div> |

TC_H_16_CS: Reserve a connector with a specific type - Amount of available connectors of a type equals the amount of reservations

| | |
|-------------------|--|
| Test case name | Reserve a connector with a specific type - Amount of available connectors of a type equals the amount of reservations |
| Test case Id | TC_H_16_CS |
| Use case Id(s) | H01(S3) |
| Requirement(s) | H01.FR.11 |
| System under test | Charging Station |
| Description | The CSMS is able to reserve an EVSE with a connector with a specific type for a specific IdToken by sending a ReserveNowRequest with a connectorType. |
| Purpose | To verify if the Charging Station is able to reserve an EVSE with a connector with a specific type, until the EV Driver with the specified IdToken arrives. |
| Prerequisite(s) | - The configuration variable ReservationCtrlr.ReservationAvailable is implemented with value <i>true</i> - The Charging Station supports the reservation of a specific connector type, that is part of the ConnectorEnumType. |

| |
|--|
| Before (Preparations) |
| Configuration State: ReservationCtrlr.ReservationEnabled is <i>true</i> (If implemented) <i>All EVSEs should be reserved</i> |
| Memory State: N/a |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|---|---|
| Charging Station | CSMS |
| 2. The Charging Station responds with a ReserveNowResponse | 1. The Test System sends a ReserveNowRequest with connectorType is <Configured connectorType> idToken.idToken <Configured valid_idtoken_idtoken> idToken.type <Configured valid_idtoken_type> |

| |
|--|
| Tool validations |
| * Step 2: Message: ReserveNowResponse - status must be <i>Occupied</i> |
| Post scenario validations: N/a |

TC_H_17_CS: Cancel reservation of an EVSE - Success

| | |
|-------------------|---|
| Test case name | Cancel reservation of an EVSE - Success |
| Test case Id | TC_H_17_CS |
| Use case Id(s) | H02 |
| Requirement(s) | H02.FR.02 |
| System under test | Charging Station |
| Description | The CSMS is able to cancel a reservation by sending a CancelReservationRequest to the Charging Station. |
| Purpose | To verify if the Charging Station is able to cancel a reservation when receiving a CancelReservationRequest from the CSMS. |
| Prerequisite(s) | The configuration variable ReservationCtrlr.ReservationAvailable is implemented with value <i>true</i> |

| |
|---|
| Before (Preparations) |
| Configuration State: ReservationCtrlr.ReservationEnabled is <i>true</i> (If implemented) |
| Memory State: <Configured evseId> is <i>Reserved</i> |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|---|---|
| Charging Station | CSMS |
| 2. The Charging Station responds with a CancelReservationResponse | 1. The Test System sends a CancelReservationRequest with reservationId is <Generated reservationId> |
| 3. The Charging Station notifies the CSMS about the status change of the connector. | 4. The Test System responds accordingly. |

| |
|--|
| Tool validations |
| <p>* Step 2:</p> <p>Message: CancelReservationResponse</p> <ul style="list-style-type: none"> - status must be <i>Accepted</i> <p>* Step 3:</p> <p>For each connector on the <Configured evseId> one of the following messages must be sent:</p> <p>Message: StatusNotificationRequest</p> <ul style="list-style-type: none"> - connectorStatus must be <i>Available</i> - evseId must be <Configured evseId> - connectorId must be <Configured connectorId> <p>Message: NotifyEventRequest</p> <ul style="list-style-type: none"> - trigger must be <i>Delta</i> - actualValue must be <i>"Available"</i> - component.name must be <i>"Connector"</i> - evse.id must be <Configured evseId> - eves.connectorId must be <Configured connectorId> - variable.name must be <i>"AvailabilityState"</i> <p>Post scenario validations:</p> <p>N/a</p> |

TC_H_18_CS: Cancel reservation of an EVSE - Rejected

| | |
|-------------------|--|
| Test case name | Cancel reservation of an EVSE - Rejected |
| Test case Id | TC_H_18_CS |
| Use case Id(s) | H02 |
| Requirement(s) | H02.FR.01 |
| System under test | Charging Station |
| Description | The CSMS is able to cancel a reservation by sending a CancelReservationRequest to the Charging Station. |
| Purpose | To verify if the Charging Station is able to reject a CancelReservationRequest , when there is no matching reservationId. |
| Prerequisite(s) | The configuration variable ReservationCtrlr.ReservationAvailable is implemented with value <i>true</i> |

Before (Preparations)

Configuration State:
ReservationCtrlr.ReservationEnabled is *true* (If implemented)

Memory State:
 N/a

Reusable State(s):
 N/a

Main (Test scenario)

| Charging Station | CSMS |
|--|---|
| 2. The Charging Station responds with a CancelReservationResponse | 1. The Test System sends a CancelReservationRequest with reservationId is 1 |

Tool validations

* Step 2:
 Message: **CancelReservationResponse**
 - **status** must be *Rejected*

Post scenario validations:
 N/a

TC_H_19_CS: Reserve a specific EVSE - Use a reserved EVSE with GroupId

| | |
|-------------------|--|
| Test case name | Reserve a specific EVSE - Use a reserved EVSE with GroupId |
| Test case Id | TC_H_19_CS |
| Use case Id(s) | H01, H03 |
| Requirement(s) | H01.FR.15,H03.FR.04,H03.FR.08 |
| System under test | Charging Station |
| Description | The CSMS is able to reserve an EVSE for a specific GroupIdToken by sending a ReserveNowRequest containing a groupIdToken . |
| Purpose | To verify if the Charging Station is able to accept an idToken with the same GroupIdToken as the idToken specified for the reservation. |
| Prerequisite(s) | The configuration variable ReservationCtrlr.ReservationAvailable is implemented with value <i>true</i> |

Before (Preparations)

Configuration State:
ReservationCtrlr.ReservationEnabled is *true* (If implemented)

Memory State:
 N/a

Reusable State(s):
 N/a

Main (Test scenario)

| Charging Station | CSMS |
|---|---|
| 2. The Charging Station responds with a ReserveNowResponse | 1. The Test System sends a ReserveNowRequest with evseld is <Configured evseld> idToken.idToken <Configured valid_idtoken_idtoken> idToken.type <Configured valid_idtoken_type> groupIdToken.idToken is <Configured groupIdToken> |
| 3. The Charging Station notifies the CSMS about the status change of the connector. <u>Note(s):</u> - The Test System expects that the Charging Station sets the <i>availabilityState</i> of the EVSE and corresponding connectors to <i>Reserved</i> . - Reporting the <i>AvailabilityState</i> of the EVSE component itself is optional. | 4. The Test System responds accordingly. |
| 3. Execute Reusable State <i>Authorized</i> | |
| <u>Note(s):</u> - <Configured valid_idtoken_idtoken2> with <Configured groupIdToken> is used for the authorization. | |
| 4. Execute Reusable State <i>EnergyTransferStarted</i> | |

| Tool validations |
|---|
| <div>* Step 2: Message: ReserveNowResponse - status must be <i>Accepted</i></div> <div>* Step 3: Message: StatusNotificationRequest - connectorStatus must be <i>Reserved</i> - evseld must be <i><Configured evseld></i> - connectorId must be <i><Configured connectorId></i> Message: NotifyEventRequest - trigger must be <i>Delta</i> - actualValue must be <i>"Reserved"</i> - component.name must be <i>"Connector"</i> - evse.id must be <i><Configured evseld></i> - eves.connectorId must be <i><Configured connectorId></i> - variable.name must be <i>"AvailabilityState"</i> (Optional) Message: NotifyEventRequest - eventData[0].trigger must be <i>Delta</i> - eventData[0].actualValue must be <i>Reserved</i> - eventData[0].component.name must be <i>EVSE</i> - eventData[0].variable.name must be <i>AvailabilityState</i></div> |
| <div>Post scenario validations: N/a</div> |

TC_H_21_CS: Charging Station cancels reservation when Unavailable

| | |
|-------------------|---|
| Test case name | Charging Station cancels reservation when Unavailable |
| Test case Id | TC_H_21_CS |
| Use case Id(s) | H01 |
| Requirement(s) | H01.FR.17 |
| System under test | Charging Station |
| Description | The Charging Station will cancel reservations, when the EVSE specified for a reservation is set to an inoperative state. |
| Purpose | To verify if the Charging Station cancels the reservation, when the availability of the EVSE specified for the reservation is set to <i>Inoperative</i> . |
| Prerequisite(s) | The configuration variable ReservationCtrlr.ReservationAvailable is implemented with value <i>true</i> |

Before (Preparations)

Configuration State:
ReservationCtrlr.ReservationEnabled is *true* (If implemented)

Memory State:
<Configured evseId> is *Reserved*

Reusable State(s):
N/a

Main (Test scenario)

| Charging Station | CSMS |
|---|---|
| 2. The Charging Station responds with a ChangeAvailabilityResponse | 1. The Test System sends a ChangeAvailabilityRequest with operationalStatus <i>Inoperative</i> and evse.id <Configured evseId> |
| 3. The Charging Station notifies the CSMS about the status change of the connector. <u>Note(s):</u> - This step needs to be executed for all connectors of the specified EVSE. - Reporting the AvailabilityState of the EVSE itself is optional. | 4. The Test System responds accordingly. |
| 5. The Charging Station sends a ReservationStatusUpdateRequest . | 6. The Test System responds with a ReservationStatusUpdateResponse . |
| 8. The Charging Station responds with a ChangeAvailabilityResponse | 7. The Test System sends a ChangeAvailabilityRequest with operationalStatus <i>Operative</i> and evse.id <Configured evseId> |
| 9. The Charging Station notifies the CSMS about the status change of the connector. <u>Note(s):</u> - This step needs to be executed for all connectors of the specified EVSE. - Reporting the AvailabilityState of the EVSE itself is optional. | 10. The Test System responds accordingly. |
| 11. Execute Reusable State <i>Authorized</i> <u>Note(s):</u> - <Configured valid idtoken fields2> are used for the authorization. | |
| 12. Execute Reusable State <i>EnergyTransferStarted</i> | |

Tool validations

* Step 2:

Message: **ChangeAvailabilityResponse**

- **status** *Accepted*

* Step 3:

Message: **StatusNotificationRequest**

- **connectorStatus** must be *Unavailable*

Message: **NotifyEventRequest**

- **eventData[0].trigger** must be *Delta*

- **eventData[0].actualValue** must be *Unavailable*

- **eventData[0].component.name** must be *Connector*

- **eventData[0].variable.name** must be *AvailabilityState*

(Reporting the AvailabilityState of the EVSE component itself is optional.)

Message: **NotifyEventRequest**

- **eventData[0].trigger** must be *Delta*

- **eventData[0].actualValue** must be *Unavailable*

- **eventData[0].component.name** must be *EVSE*

- **eventData[0].variable.name** must be *AvailabilityState*

* Step 5:

Message: **ReservationStatusUpdateRequest**

- **reservationId** must be *<Generated reservationId>*

- **reservationUpdateStatus** must be *Removed*

* Step 8:

Message: **ChangeAvailabilityResponse**

- **status** *Accepted*

* Step 9:

Message: **StatusNotificationRequest**

- **connectorStatus** must be *Available*

Message: **NotifyEventRequest**

- **eventData[0].trigger** must be *Delta*

- **eventData[0].actualValue** must be *Available*

- **eventData[0].component.name** must be *Connector*

- **eventData[0].variable.name** must be *AvailabilityState*

(Reporting the AvailabilityState of the EVSE component itself is optional.)

Message: **NotifyEventRequest**

- **eventData[0].trigger** must be *Delta*

- **eventData[0].actualValue** must be *Available*

- **eventData[0].component.name** must be *EVSE*

- **eventData[0].variable.name** must be *AvailabilityState*

Post scenario validations:

N/a

TC_H_22_CS: Reserve a specific EVSE - Configured to Reject

| | |
|-------------------|---|
| Test case name | Reserve a specific EVSE - Configured to Reject |
| Test case Id | TC_H_22_CS |
| Use case Id(s) | H01 |
| Requirement(s) | H01.FR.01 |
| System under test | Charging Station |
| Description | The CSMS is able to reserve a specific EVSE for a specific IdToken by sending a ReserveNowRequest containing an evseld. |
| Purpose | To verify if the Charging Station is able to correctly respond when it is configured not to accept reservations. |
| Prerequisite(s) | The configuration variable ReservationCtrlr.ReservationAvailable is implemented with value <i>false</i> |

| |
|--|
| Before (Preparations) |
| Configuration State: ReservationCtrlr.ReservationEnabled is <i>false</i> (If implemented) |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a ReserveNowResponse | 1. The Test System sends a ReserveNowRequest |

| |
|--|
| Tool validations |
| * Step 2: Message: ReserveNowResponse - status <i>Rejected</i> |
| Post scenario validations: N/a |

TC_H_23_CS: Reserve a specific EVSE - Replace reservation

| | |
|-------------------|--|
| Test case name | Reserve a specific EVSE - Replace reservation |
| Test case Id | TC_H_23_CS |
| Use case Id(s) | H01 |
| Requirement(s) | H01.FR.02 |
| System under test | Charging Station |
| Description | The CSMS is able to reserve a specific EVSE for a specific IdToken by sending a ReserveNowRequest containing an evseld. |
| Purpose | To verify if the Charging Station is able to replace a reservation of a specific EVSE, until the EV Driver with the specified IdToken arrives. |
| Prerequisite(s) | The configuration variable ReservationCtrlr.ReservationAvailable is implemented with value <i>true</i> |

Before (Preparations)

Configuration State:

ReservationCtrlr.ReservationEnabled is *true* (If implemented)

Memory State:

A reservation is valid on <Configured evseld> with <Configured valid_idtoken>

Reusable State(s):

N/a

Main (Test scenario)

| Charging Station | CSMS |
|---|--|
| 2. The Charging Station responds with a ReserveNowResponse | 1. The Test System sends a ReserveNowRequest with id <Configured reservationId> evseld is <Configured evseld> idToken.idToken <Configured valid_idtoken_idtoken2> idToken.type <Configured valid_idtoken_type2> |
| 3. Execute Reusable State <i>Authorized</i> | |
| <u>Note(s):</u> - <Configured valid idToken2 fields> are used for the authorization. | |
| 4. Execute Reusable State <i>EnergyTransferStarted</i> | |

| Tool validations |
|--|
| <p>* Step 2:</p> <p>Message: ReserveNowResponse</p> <ul style="list-style-type: none">- status must be <i>Accepted</i> <p>* Step 3:</p> <p>Message: StatusNotificationRequest</p> <ul style="list-style-type: none">- connectorStatus must be <i>Reserved</i>- evseld must be <i><Specified evseld></i>- connectorId must be <i><Configured connectorId></i> <p>Message: NotifyEventRequest</p> <ul style="list-style-type: none">- trigger must be <i>Delta</i>- actualValue must be <i>"Reserved"</i>- component.name must be <i>"Connector"</i>- evse.id must be <i><Specified evseld></i>- eves.connectorId must be <i><Configured connectorId></i>- variable.name must be <i>"AvailabilityState"</i> <p>(Optional)</p> <p>Message: NotifyEventRequest</p> <ul style="list-style-type: none">- eventData[0].trigger must be <i>Delta</i>- eventData[0].actualValue must be <i>Reserved</i>- eventData[0].component.name must be <i>EVSE</i>- eventData[0].variable.name must be <i>AvailabilityState</i> |
| <p>Post scenario validations:</p> <p>N/a</p> |

TC_H_24_CS: Reserve an unspecified EVSE - GroupIdToken

| | |
|-------------------|---|
| Test case name | Reserve an unspecified EVSE - GroupIdToken |
| Test case Id | TC_H_24_CS |
| Use case Id(s) | H03 |
| Requirement(s) | H03.FR.06 |
| System under test | Charging Station |
| Description | The CSMS is able to reserve a specific EVSE for a specific IdToken by sending a ReserveNowRequest containing an evseld. |
| Purpose | To verify if the Charging Station is able to reserve a unspecific EVSE, until the EV Driver with the specified groupIdToken arrives. Since all other EVSEs are reserved, this reservation will force the last EVSE to also become reserved. |
| Prerequisite(s) | The configuration variable ReservationCtrlr.ReservationAvailable is implemented with value <i>true</i> |

| |
|--|
| Before (Preparations) |
| Configuration State: ReservationCtrlr.ReservationEnabled is <i>true</i> (If implemented) |
| Memory State: N/a |
| Reusable State(s): All EVSEs except one are <i>Reserved</i> |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a ReserveNowResponse | 1. The Test System sends a ReserveNowRequest with idToken.idToken is <i><Configured valid_idtoken></i> groupIdToken.idToken is <i><Configured group_idtoken></i> |
| 3. The Charging Station notifies the CSMS about the status change of the connector. <u>Note(s):</u> - The Test System expects that the Charging Station sets the <i>availabilityState</i> of the only not reserved EVSE and corresponding connectors to <i>Reserved</i> . - Reporting the <i>AvailabilityState</i> of the EVSE component itself is optional. | 4. The Test System responds accordingly. |
| 5. Execute Reusable State <i>Authorized</i> <u>Note(s):</u> - <i><Configured valid_idtoken2></i> is used for the authorization. | |
| 6. Execute Reusable State <i>EnergyTransferStarted</i> | |

| Tool validations |
|--|
| <div>* Step 2: Message: ReserveNowResponse - status must be <i>Accepted</i></div> <div>* Step 3: Message: StatusNotificationRequest - connectorStatus must be <i>Reserved</i></div> <div>Message: NotifyEventRequest - eventData[0].trigger must be <i>Delta</i> - eventData[0].actualValue must be <i>Reserved</i> - eventData[0].component.name must be <i>Connector</i> - eventData[0].variable.name must be <i>AvailabilityState</i> (Optional) Message: NotifyEventRequest - eventData[0].trigger must be <i>Delta</i> - eventData[0].actualValue must be <i>Reserved</i> - eventData[0].component.name must be <i>EVSE</i> - eventData[0].variable.name must be <i>AvailabilityState</i></div> |
| <div>Post scenario validations: N/a</div> |

I Tariff and Cost

TC_I_01_CS: Show EV Driver running total cost during charging - costUpdatedRequest

| | |
|-------------------|---|
| Test case name | Show EV Driver running total cost during charging - costUpdatedRequest |
| Test case Id | TC_I_01_CS |
| Use case Id(s) | I02 |
| Requirement(s) | I02.FR.02 |
| System under test | Charging Station |
| Description | While a transaction is ongoing, the driver wants to know how much the running total cost is, updated at a relevant interval. |
| Purpose | To verify if the Charging Station is able to correctly display the running total cost as described in the OCPP specification. |
| Prerequisite(s) | - The Charging Station supports Tariff Information |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State(s):

State is *EVConnectedPreSession*

Main (Test scenario)

| | |
|---|--|
| Charging Station | CSMS |
| <u>Manual Action:</u> Present valid idToken | |
| 1. The Charging Station sends an AuthorizeRequest | 2. The Test System responds with an AuthorizeResponse with - idTokenInfo.status <i>Accepted</i> - idTokenInfo.personalMessage.content <Configured Cost> |
| 3. The Charging Station sends a TransactionEventRequest <u>Note(s):</u> - This step needs to be executed when TxStartPoint contains Authorized OR the transaction already started. So in the case TxStartPoint contains <i>ParkingBayOccupancy</i> | 4. The Test System responds with a TransactionEventResponse with - idTokenInfo.status <i>Accepted</i> |
| 5. Execute Reusable State <i>EnergyTransferStarted</i> | |
| 6. The Charging Station sends an TransactionEventRequest | 7. The Test System responds with an TransactionEventResponse with - updatedPersonalMessage.content <Configured Cost> |
| 9. The Charging Station responds with a CostUpdatedResponse | 8. The Test System sends a CostUpdatedRequest with - totalCost <Configured Cost2> - transactionId <Configured transactionId> |
| <u>Note(s):</u> Step 6, 7, 8, and 9 are repeated <i>n</i> times | |

| Tool validations |
|--|
| <div>* Step 1: Message AuthorizeRequest - idToken.idToken <Configured valid_idtoken_idtoken> - idToken.type <Configured valid_idtoken_type> * Step 3: Message TransactionEventRequest - triggerReason <i>Authorized</i> - idToken.idToken <Configured valid_idtoken_idtoken> - idToken.type <Configured valid_idtoken_type> if transaction was already started - eventType <i>Updated</i> else - eventType <i>Started</i></div> |
| <div>Post scenario validations: - N/a</div> |

TC_I_02_CS: Show EV Driver Final Total Cost After Charging

| | |
|-------------------|--|
| Test case name | Show EV Driver Final Total Cost After Charging |
| Test case Id | TC_I_02_CS |
| Use case Id(s) | I03 |
| Requirement(s) | I03.FR.01, I03.FR.03 |
| System under test | Charging Station |
| Description | While a transaction is ongoing, the driver wants to know how much the running total cost is, updated at a relevant interval. |
| Purpose | To verify if the Charging Station is able to correctly display the total cost as described in the OCPP specification. |
| Prerequisite(s) | - The Charging Station supports Tariff Information |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> |

| Main (Test scenario) | |
|---|------|
| Charging Station | CSMS |
| <u>Manual Action</u> : <i>Present valid idToken</i> | |
| 1. Execute Reusable State <i>StopAuthorized</i> <u>Note</u> : IF Message TransactionEventRequest - eventType <i>Ended</i> THEN Message TransactionEventResponse - totalCost <i><Generated Cost></i> | |
| 2. Execute Reusable State <i>EVConnectedPostSession</i> <u>Note</u> : IF Message TransactionEventRequest - eventType <i>Ended</i> THEN Message TransactionEventResponse - totalCost <i><Generated Cost></i> | |
| 3. Execute Reusable State <i>EVDIsconnected</i> <u>Note</u> : IF Message TransactionEventRequest - eventType <i>Ended</i> THEN Message TransactionEventResponse - totalCost <i><Generated Cost></i> | |
| 4. Execute Reusable State <i>ParkingBayUnoccupied</i> <u>Note</u> : IF Message TransactionEventRequest - eventType <i>Ended</i> THEN Message TransactionEventResponse - totalCost <i><Generated Cost></i> | |

| |
|-------------------------------------|
| Tool validations |
| N/a |
| Post scenario validations: - N/a |

TC_I_07_CS: Show EV Driver running total cost during charging - transactionEventResponse

| | |
|-------------------|---|
| Test case name | Show EV Driver running total cost during charging - transactionEventResponse |
| Test case Id | TC_I_07_CS |
| Use case Id(s) | I02 |
| Requirement(s) | I02.FR.02 |
| System under test | Charging Station |
| Description | While a transaction is ongoing, the driver wants to know how much the running total cost is, updated at a relevant interval. |
| Purpose | To verify if the Charging Station is able to correctly display the running total cost as described in the OCPP specification. |
| Prerequisite(s) | - The Charging Station supports Tariff Information |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EVConnectedPreSession</i> |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action:</u> <i>Present valid idToken</i> | |
| 1. The Charging Station sends an AuthorizeRequest | 2. The Test System responds with an AuthorizeResponse with - idTokenInfo.status <i>Accepted</i> - idTokenInfo.personalMessage.content <i><Configured Cost></i> |
| 3. The Charging Station sends a TransactionEventRequest <u>Note(s):</u> - <i>This step needs to be executed when TxStartPoint contains Authorized OR the transaction already started. So in the case TxStartPoint contains ParkingBayOccupancy</i> | 4. The Test System responds with a TransactionEventResponse with - idTokenInfo.status <i>Accepted</i> |
| 5. Execute Reusable State <i>EnergyTransferStarted</i> | |
| 6. The Charging Station sends an TransactionEventRequest | 7. The Test System responds with an TransactionEventResponse with - updatedPersonalMessage.content <i><Configured Cost></i> |
| 9. The Charging Station responds with a TransactionEventResponse | 8. The Test System sends a TransactionEventRequest with - totalCost <i><Configured Cost2></i> - transactionId <i><Configured transactionId></i> |
| <u>Note(s):</u> <i>Step 6, 7, 8, and 9 are repeated n times</i> | |

| Tool validations |
|--|
| <div>* Step 1: Message AuthorizeRequest - idToken.idToken <Configured valid_idtoken_idtoken> - idToken.type <Configured valid_idtoken_type> * Step 3: Message TransactionEventRequest - triggerReason <i>Authorized</i> - idToken.idToken <Configured valid_idtoken_idtoken> - idToken.type <Configured valid_idtoken_type> if transaction was already started - eventType <i>Updated</i> else - eventType <i>Started</i></div> |
| <div>Post scenario validations: - N/a</div> |

J Meter Values

TC_J_01_CS: Clock-aligned Meter Values - No transaction ongoing

| | |
|-------------------|--|
| Test case name | Clock-aligned Meter Values - No transaction ongoing |
| Test case Id | TC_J_01_CS |
| Use case Id(s) | J01 |
| Requirement(s) | J01.FR.01,J01.FR.02,J01.FR.03,J01.FR.06,J01.FR.07,J01.FR.08,J01.FR.14,J01.FR.15 |
| System under test | Charging Station |
| Description | The Charging Station samples the electrical meter or other sensor/transducer hardware to provide information about its Meter Values. Depending on configuration settings, the Charging Station will send Meter Values. |
| Purpose | To verify if the Charging Station is able to send clock-aligned Meter Values, when it is configured to do so. |
| Prerequisite(s) | The Charging Station has an energy meter. |

| |
|---|
| Before (Preparations) |
| Configuration State: AlignedDataInterval is <Configured clock-aligned Meter Values interval> |
| Memory State: N/a |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|---|---|
| Charging Station | CSMS |
| <p>1. The Charging Station notifies the CSMS about its measured Meter Values.</p> <p><u>Note(s):</u></p> <ul style="list-style-type: none">- The Meter Value messages do NOT need to be send/received at the exact specified interval. The configured measurands must be measured at the configured interval.- Multiple Meter Value messages may be sent per configured interval. One (or more in case the amount of measured data is too much for one message) for each EVSE and one (or more) for the main power meter (connectorId=0)- The Test System will end the testcase after it has received three Meter Value messages. | <p>2. The Test System responds accordingly.</p> |

| |
|---|
| Tool validations |
| * Step 1: Message: MeterValuesRequest - sampledValue[0].context must be <i>Sample.Clock</i> - sampledValue must contain <An element per configured measurand at the <i>AlignedDataMeasurands</i> . The measurand field may be omitted when the measurand is "Energy.Active.Import.Register"> |
| Post scenario validations: Message: MeterValuesRequest - timestamp <The intervals between the timestamps of the received Meter Value messages must equal the configured value at <i>AlignedDataInterval</i> . However it is allowed to send multiple Meter Value messages per configured interval. One (or more in case the amount of measured data is too much for one message) for each EVSE and one (or more) for the main power meter (evseld=0). But the timestamp of these messages must all be the same.> - None of the provided sampledValues shall have location = EV, except when measurand = SoC. |

TC_J_02_CS: Clock-aligned Meter Values - Transaction ongoing

| | |
|-------------------|--|
| Test case name | Clock-aligned Meter Values - Transaction ongoing |
| Test case Id | TC_J_02_CS |
| Use case Id(s) | J01 |
| Requirement(s) | J01.FR.01,J01.FR.02,J01.FR.03,J01.FR.06,J01.FR.07,J01.FR.08,J01.FR.14,J01.FR.15 |
| System under test | Charging Station |
| Description | The Charging Station samples the electrical meter or other sensor/transducer hardware to provide information about its Meter Values. Depending on configuration settings, the Charging Station will send Meter Values. |
| Purpose | To verify if the Charging Station is able to send clock-aligned Meter Values, while a transaction is ongoing, when it is configured to do so. |
| Prerequisite(s) | The Charging Station has an energy meter. |

| |
|---|
| Before (Preparations) |
| Configuration State: AlignedDataInterval is <Configured clock-aligned Meter Values interval> AlignedDataSendDuringIdle is <i>false</i> (If implemented) RegisterValuesWithoutPhases is <i>false</i> (If implemented) |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> |

| Main (Test scenario) | |
|--|---|
| Charging Station | CSMS |
| <u>Note(s):</u> - The Charging Station can follow Steps 1 and 2 or Steps 3 and 4 | |
| 1. The Charging Station notifies the CSMS about its measured Meter Values. <u>Note(s):</u> - During a transaction the MeterValueRequest can still be used to report meter values for the main power meter (evseld=0) and idle EVSEs - The Meter Value messages do NOT need to be send/received at the exact specified interval. The configured measurands must be measured at the configured interval. - Multiple Meter Value messages may be sent per configured interval, in case the amount of measured data is too much for one message. | 2. The Test System responds accordingly. |
| 3. The Charging Station sends a TransactionEventRequest <u>Note(s):</u> - During a transaction the meter values for the configured EVSE with the ongoing transaction should be transmitted using the TransactionEventRequest. - The TransactionEventRequest messages do NOT need to be send/received at the exact specified interval. The configured measurands must be measured at the configured interval. - Multiple TransactionEventRequest messages may be sent per configured interval, in case the amount of measured data is too much for one message. - The Test System will end the testcase after it has the <Configured transaction duration> is reached. | 4. The Test System responds with a TransactionEventResponse |

Tool validations

Note: The following steps do not need to be sent in a specific order.

* Step 1:

Message: **MeterValuesRequest**

- **meterValue[0].sampledValue[0].context** must be *Sample.Clock*
- **meterValue[0].sampledValue** must contain <An element per configured measurand at the *AlignedDataMeasurands*>

Notes:

- The measurand field may be omitted when the measurand is "Energy.Active.Import.Register"
- It is possible that measurands are reported on multiple locations or phases, based on the capabilities of the energy meter.

* Step 3:

Message: **TransactionEventRequest**

- **triggerReason** must be *MeterValueClock*
- **meterValue[0].sampledValue[0].context** must be *Sample.Clock*
- **meterValue[0].sampledValue** must contain <An element per configured measurand at the *AlignedDataMeasurands*>

Notes:

- The measurand field may be omitted when the measurand is "Energy.Active.Import.Register"
- It is possible that measurands are reported on multiple locations or phases, based on the capabilities of the energy meter.

Post scenario validations:

Message: **TransactionEventRequest**

- **timestamp** <The intervals between the timestamps of the received *TransactionEventRequest* messages must equal the configured value at *AlignedDataInterval*. However it is allowed to send multiple Meter Value messages per configured interval, in case the amount of measured data is too much for one message. But the timestamp of these messages must all be the same.>

Message: **MeterValuesRequest**

- **timestamp** <The intervals between the timestamps of the received Meter Value messages must equal the configured value at *AlignedDataInterval*. However it is allowed to send multiple Meter Value messages per configured interval, in case the amount of measured data is too much for one message. But the timestamp of these messages must all be the same.>
- None of the provided **sampledValues** shall have **location** = EV, except when **measurand** = SoC.

TC_J_03_CS: Clock-aligned Meter Values - EventType Ended

| | |
|-------------------|--|
| Test case name | Clock-aligned Meter Values - EventType Ended |
| Test case Id | TC_J_03_CS |
| Use case Id(s) | J01 & (E06,E07,E08,E09,E10,E12) |
| Requirement(s) | J01.FR.01,J01.FR.02,J01.FR.03,J01.FR.06,J01.FR.07,J01.FR.08,J01.FR.14,J01.FR.15, E06.FR.11,E06.FR.17,E07.FR.08,E07.FR.13,E08.FR.09,E09.FR.05,E10.FR.04,E12.FR.07 |
| System under test | Charging Station |
| Description | The Charging Station samples the electrical meter or other sensor/transducer hardware to provide information about its Meter Values. Depending on configuration settings, the Charging Station will send Meter Values. |
| Purpose | To verify if the Charging Station is able to send clock-aligned Meter Values when a transaction ends at the TransactionEventRequest with eventType is <i>Ended</i> , when it is configured to do so. |
| Prerequisite(s) | The Charging Station has an energy meter. |

Before (Preparations)

Configuration State:

AlignedDataTxEndedInterval is *<Configured clock_aligned_tx_ended_meter_values_interval>*

SampledDataTxEndedMeasurands is *empty string*

AlignedDataSendDuringIdle is *false* (If implemented)

Memory State:

N/a

Reusable State(s):

State is *EnergyTransferStarted*

Main (Test scenario)

| | |
|--|------|
| Charging Station | CSMS |
| 1. Execute Reusable State <i>ParkingBayUnoccupied</i> | |
| <p><u>Note(s):</u></p> <ul style="list-style-type: none"> - This step will be executed after the <i><Configured transaction duration></i> is reached. - This causes the transaction to stop. | |

Tool validations

N/a

Post scenario validations:

- The **TransactionEventRequest** containing eventType *Ended* contains **meterValue** fields.
- The **meterValue** must contain an array of *<elements per data collection moment indicated by AlignedDataTxEndedInterval>*.
- **metervalue[0].timestamp** must be greater/equal then start time of transaction.
- *<The intervals between the timestamps of the meterValues must equal the configured value at AlignedDataTxEndedInterval.>*
- Last **metervalue[n].timestamp** must be smaller/equal then end time of transaction.
- **sampledValue[x].context** must be *Sample.Clock* for all **sampledValues**.
- **sampledValue** must contain *<An element per configured measurand at the AlignedDataTxEndedMeasurands>*. The measurand field may be omitted when the measurand is "Energy.Active.Import.Register">_
- None of the provided **sampledValues** shall have **location** = EV, except when **measurand** = SoC.

TC_J_04_CS: Clock-aligned Meter Values - Signed

| | |
|-------------------|--|
| Test case name | Clock-aligned Meter Values - Signed |
| Test case Id | TC_J_04_CS |
| Use case Id(s) | J01 |
| Requirement(s) | J01.FR.21 |
| System under test | Charging Station |
| Description | The Charging Station samples the electrical meter or other sensor/transducer hardware to provide information about its Meter Values. Depending on configuration settings, the Charging Station will send Meter Values. |
| Purpose | To verify if the Charging Station is able to send signed clock-aligned Meter Values when a transaction ends at the TransactionEventRequest with eventType is <i>Ended</i> , when it is configured to do so. |
| Prerequisite(s) | The Charging Station has an energy meter. |

| |
|---|
| Before (Preparations) |
| Configuration State: AlignedDataTxEndedInterval is <Configured clock_aligned_tx_ended_meter_values_interval> AlignedDataSendDuringIdle is <i>false</i> (If implemented) AlignedDataSignReadings is <i>true</i> |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> |

| | |
|--|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>ParkingBayUnoccupied</i> | |
| <u>Note(s)</u> : - This step will be executed after the <Configured transaction duration> is reached. - This causes the transaction to stop. | |

| |
|---|
| Tool validations |
| N/a |
| Post scenario validations: - The TransactionEventRequest containing eventType <i>Ended</i> contains the MeterValue field. - The MeterValue should contain <An element per data collection moment indicated by AlignedDataTxEndedInterval. The Test System will not validate this.> - timestamp <The intervals between the timestamps of the received Meter Value messages should equal the configured value at AlignedDataTxEndedInterval.> - sampledValue[0].context should be <i>Sample.Clock</i> - sampledValue should contain <An element per configured measurand at the AlignedDataTxEndedMeasurands. The measurand field may be omitted when the measurand is "Energy.Active.Import.Register"> - sampledValue.signedMeterValue should not be omitted - sampledValue.signedMeterValue.publicKey should exist and depending on the value of OCPPPCommCtrlr.PublicKeyWithSignedMeterValue, should be either "", or a valid public key - None of the provided sampledValues shall have location = <i>EV</i> , except when measurand = <i>SoC</i> . |

TC_J_06_CS: Clock-aligned Meter Values - No Meter Values during transaction

| | |
|-------------------|--|
| Test case name | Clock-aligned Meter Values - No Meter Values during transaction |
| Test case Id | TC_J_06_CS |
| Use case Id(s) | J01 |
| Requirement(s) | N/a |
| System under test | Charging Station |
| Description | The Charging Station samples the electrical meter or other sensor/transducer hardware to provide information about its Meter Values. Depending on configuration settings, the Charging Station will send Meter Values. |
| Purpose | To verify if the Charging Station is able to only send clock-aligned Meter Values when there is no ongoing transaction, when it is configured to do so. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The Charging Station has an energy meter. - The configuration variable AlignedDataSendDuringIdle is implemented. |

Before (Preparations)

Configuration State:

AlignedDataInterval is set to <Configured clock-aligned Meter Values interval>

AlignedDataSendDuringIdle is set to *true*

Memory State:

N/a

Reusable State(s):

N/a

Main (Test scenario)

| Charging Station | CSMS |
|---|---|
| <p>1. The Charging Station notifies the CSMS about its measured Meter Values.</p> <p><u>Note(s):</u></p> <ul style="list-style-type: none"> - The Meter Value messages do NOT need to be send/received at the exact specified interval. The configured measurands must be measured at the configured interval. - Multiple Meter Value messages may be sent per configured interval. One (or more in case the amount of measured data is too much for one message) for each EVSE and one (or more) for the main power meter (evseld=0) | <p>2. The Test System responds accordingly.</p> |
| <p>3. Execute Reusable State <i>EnergyTransferStarted</i></p> | |
| <p>4. The Charging Station notifies the CSMS about its measured Meter Values.</p> <p><u>Note(s):</u></p> <ul style="list-style-type: none"> - The Meter Value messages should not be send/received at the exact specified interval. | <p>5. The Test System responds accordingly.</p> |
| <p>6. Execute Reusable State <i>ParkingBayUnoccupied</i></p> <p><u>Note(s):</u></p> <ul style="list-style-type: none"> - This step will be executed after the <Configured clock-aligned Meter Values interval + 5 seconds> is reached. | |

| Main (Test scenario) | |
|---|---|
| <p>7. The Charging Station notifies the CSMS about its measured Meter Values.</p> <p><u>Note(s):</u></p> <ul style="list-style-type: none"> - The Meter Value messages do NOT need to be send/received at the exact specified interval. The configured measurands must be measured at the configured interval. - Multiple Meter Value messages may be sent per configured interval. One (or more in case the amount of measured data is too much for one message) for each EVSE and one (or more) for the main power meter (evseld=0) | <p>8. The Test System responds accordingly.</p> |

| Tool validations |
|---|
| <p>* Step 1 & 7:</p> <p>Message: MeterValuesRequest</p> <ul style="list-style-type: none"> - sampledValue[0].context must be <i>Sample.Clock</i> - sampledValue must contain <An element per configured measurand at the <i>AlignedDataMeasurands</i>. The measurand field may be omitted when the measurand is "Energy.Active.Import.Register"> |
| <p>Post scenario validations:</p> <p>Message: MeterValuesRequest</p> <ul style="list-style-type: none"> - timestamp <The intervals between the timestamps of the received Meter Value messages must equal the configured value at <i>AlignedDataInterval</i>. However it is allowed to send multiple Meter Value messages per configured interval. One (or more in case the amount of measured data is too much for one message) for each EVSE and one (or more) for the main power meter (evseld=0). But the timestamp of these messages must all be the same.> - The Charging Station did not send any message to report Meter Values to the Test System, during the time the transaction was active at step 3 and 4. This means none of the following; MeterValuesRequest OR TransactionEventRequest containing the MeterValue field. - None of the provided sampledValues shall have location = EV, except when measurand = SoC. |

TC_J_07_CS: Sampled Meter Values - EventType Started - EVSE known

| Test case name | Sampled Meter Values - EventType Started - EVSE known |
|-------------------|---|
| Test case Id | TC_J_07_CS |
| Use case Id(s) | J02 & (E01,E02,E03,E09,E04,E05) |
| Requirement(s) | J02.FR.01,J02.FR.02,J02.FR.03,J02.FR.04,J02.FR.10,E01.FR.09,E02.FR.09,E03.FR.07,E04.FR.05,E05.FR.05 |
| System under test | Charging Station |
| Description | The Charging Station samples the electrical meter or other sensor/transducer hardware to provide information about its Meter Values. Depending on configuration settings, the Charging Station will send Meter Values. |
| Purpose | To verify if the Charging Station is able to send sampled Meter Values when a transaction starts and the EVSE is known, at the TransactionEventRequest with eventType is <i>Started</i> , when it is configured to do so. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The Charging Station has an energy meter. - The Charging Station does NOT have the following configuration; TxStartPoint contains <i>ParkingBayOccupancy</i> |

| Before (Preparations) |
|--|
| Configuration State: TxStartPoint contains <i>EVConnected</i> <u>Note:</u> TxStartPoint contains <i>EVConnected</i> , <i>Authorized</i> , <i>PowerPathClosed</i> , <i>EnergyTransfer</i> AND/OR <i>DataSigned</i> (At least one of these values must be set). |
| Memory State: N/a |
| Reusable State(s): State is <i>ParkingBayOccupied</i> |

| Main (Test scenario) | |
|---|------|
| Charging Station | CSMS |
| 1. Execute Reusable State <i>EVConnectedPreSession</i> | |
| 2. Execute Reusable State <i>EnergyTransferStarted</i> | |

| Tool validations |
|--|
| N/a |
| Post scenario validations: <ul style="list-style-type: none"> - The TransactionEventRequest containing eventType <i>Started</i> contains the MeterValue field. - sampledValue[0].context must be <i>Transaction.Begin</i> - sampledValue must contain <An element per configured measurand at the SampledDataTxStartedMeasurands. The measurand field may be omitted when the measurand is "Energy.Active.Import.Register"> - None of the provided sampledValues shall have location = EV, except when measurand = SoC. |

TC_J_08_CS: Sampled Meter Values - Context Transaction.Begin - EVSE not known

| | |
|-------------------|---|
| Test case name | Sampled Meter Values - Context Transaction.Begin - EVSE not known |
| Test case Id | TC_J_08_CS |
| Use case Id(s) | J02 & (E01,E02,E03,E09,E04,E05) |
| Requirement(s) | J02.FR.01, J02.FR.02, J02.FR.03, J02.FR.04, J02.FR.10, E01.FR.16, E01.FR.17, E03.FR.11, E04.FR.11, E05.FR.08 |
| System under test | Charging Station |
| Description | The Charging Station samples the electrical meter or other sensor/transducer hardware to provide information about its Meter Values. Depending on configuration settings, the Charging Station will send Meter Values. |
| Purpose | To verify if the Charging Station sends Meter Values for Transaction.Begin as soon as the EVSE to be used is known, for a transaction that starts before the cable is plugged in. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The Charging Station has an energy meter. - The Charging Station does NOT have the following configuration; TxStartPoint does NOT contain <i>ParkingBayOccupancy</i> OR <i>Authorized</i>. |

| |
|--|
| Before (Preparations) |
| Configuration State: TxStartPoint contains <i>Authorized</i> Note: TxStartPoint contains <i>Authorized</i> AND/OR <i>ParkingBayOccupancy</i> (At least one of these values must be set). |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>EnergyTransferStarted</i> | |

| |
|---|
| Tool validations |
| N/a |
| Post scenario validations: <ul style="list-style-type: none"> - The first TransactionEventRequest containing a value for evse, sent during the execution of reusable state <i>EVConnectedPreSession</i> contains the MeterValue field with: <ul style="list-style-type: none"> - sampledValue[0].context must be <i>Transaction.Begin</i> - sampledValue must contain <An element per configured measurand at the <i>SampledDataTxStartedMeasurands</i>. The measurand field may be omitted when the measurand is "Energy.Active.Import.Register"> - None of the provided sampledValues shall have location = EV, except when measurand = SoC. |

TC_J_09_CS: Sampled Meter Values - EventType Updated

| | |
|-------------------|--|
| Test case name | Sampled Meter Values - EventType Updated |
| Test case Id | TC_J_09_CS |
| Use case Id(s) | J02 & (E01,E02,E03,E09,E04,E05) |
| Requirement(s) | J02.FR.01, J02.FR.02, J02.FR.03, J02.FR.04, J02.FR.10, J02.FR.11, J02.FR.14, E02.FR.10, E02.FR.11, E03.FR.08, E03.FR.09, E04.FR.06, E04.FR.09, E11.FR.03, E11.FR.06, E12.FR.03, E12.FR.06 |
| System under test | Charging Station |
| Description | The Charging Station samples the electrical meter or other sensor/transducer hardware to provide information about its Meter Values. Depending on configuration settings, the Charging Station will send Meter Values. |
| Purpose | To verify if the Charging Station is able to send sampled Meter Values during the transaction, at the TransactionEventRequest with eventType is <i>Updated</i> , when it is configured to do so. |
| Prerequisite(s) | The Charging Station has an energy meter. |

| |
|--|
| Before (Preparations) |
| Configuration State: SampledDataTxUpdatedInterval is <Configured sampled Meter Values Updated interval> |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> |

| Main (Test scenario) | |
|--|---|
| Charging Station | CSMS |
| <p>1. The Charging Station sends a TransactionEventRequest</p> <p><u>Note(s):</u></p> <ul style="list-style-type: none">- The <i>TransactionEventRequest</i> messages do NOT need to be send/received at the exact specified interval. The configured measurands must be measured at the configured interval.- Multiple <i>TransactionEventRequest</i> messages may be sent per configured interval, in case the amount of measured data is too much for one message._ - The Test System will end the testcase after it has the _<Configured transaction duration> is reached._ | <p>2. The Test System responds with a TransactionEventResponse</p> |

| |
|--|
| Tool validations |
| * Step 1: Message: TransactionEventRequest - triggerReason must be <i>MeterValuePeriodic</i> - sampledValue[0].context must be <i>Sample.Periodic</i> - sampledValue must contain <An element per configured measurand at the SampledDataTxUpdatedMeasurands. The measurand field may be omitted when the measurand is "Energy.Active.Import.Register"> |
| Post scenario validations: - timestamp <The intervals between the timestamps of the received TransactionEventRequest messages must equal the configured value at SampledDataTxUpdatedInterval. However it is allowed to send multiple Meter Value messages per configured interval, in case the amount of measured data is too much for one message. But the timestamp of these messages must all be the same.> - None of the provided sampledValues shall have location = EV, except when measurand = SoC. |

TC_J_10_CS: Sampled Meter Values - EventType Ended

| | |
|-------------------|--|
| Test case name | Sampled Meter Values - EventType Ended |
| Test case Id | TC_J_10_CS |
| Use case Id(s) | J02 & (E06,E07,E08,E09,E10,E12) |
| Requirement(s) | J02.FR.01,J02.FR.02,J02.FR.03,J02.FR.04,J02.FR.10,E06.FR.11, E07.FR.08,E08.FR.09,E09.FR.05,E10.FR.04,E12.FR.07 |
| System under test | Charging Station |
| Description | The Charging Station samples the electrical meter or other sensor/transducer hardware to provide information about its Meter Values. Depending on configuration settings, the Charging Station will send Meter Values. |
| Purpose | To verify if the Charging Station is able to send sampled Meter Values when a transaction ends at the TransactionEventRequest with eventType is <i>Ended</i> , when it is configured to do so. |
| Prerequisite(s) | The Charging Station has an energy meter. |

| |
|--|
| Before (Preparations) |
| Configuration State: SampledDataTxEndedInterval is <i><Configured sampled_tx_ended_meter_values_interval></i> AlignedDataTxEndedMeasurands is <i>empty string</i> |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> |

| | |
|---|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>ParkingBayUnoccupied</i> | |
| <u>Note(s):</u> - This step will be executed after the <Configured transaction duration> is reached. - This causes the transaction to stop. | |

| |
|--|
| Tool validations |
| N/a |
| Post scenario validations: - The TransactionEventRequest containing eventType <i>Ended</i> contains the MeterValue field. - The MeterValue must contain <i><An element per data collection moment indicated by SampledDataTxEndedInterval. The Test System will not validate this.></i> - timestamp <i><The intervals between the timestamps of the received Meter Value messages must equal the configured value at SampledDataTxEndedInterval.></i> - sampledValue[0].context must be <i>Sample.Periodic</i> AND one must have <i>Transaction.End</i> - sampledValue must contain <i><An element per configured measurand at the SampledDataTxEndedMeasurands. The measurand field may be omitted when the measurand is "Energy.Active.Import.Register"></i> - None of the provided sampledValues shall have location = EV, except when measurand = SoC. |

TC_J_11_CS: Sampled Meter Values - Signed

| | |
|-------------------|--|
| Test case name | Sampled Meter Values - Signed |
| Test case Id | TC_J_11_CS |
| Use case Id(s) | J02 |
| Requirement(s) | J02.FR.21 |
| System under test | Charging Station |
| Description | The Charging Station samples the electrical meter or other sensor/transducer hardware to provide information about its Meter Values. Depending on configuration settings, the Charging Station will send Meter Values. |
| Purpose | To verify if the Charging Station is able to send sampled Meter Values when a transaction ends at the TransactionEventRequest with eventType is <i>Ended</i> , when it is configured to do so. |
| Prerequisite(s) | The Charging Station has an energy meter. |

Before (Preparations)

Configuration State:

SampledDataTxEndedInterval is *<Configured sampled_tx_ended_meter_values_interval>*

SampledDataSignReadings is true

Memory State:

N/a

Reusable State(s):

State is *EnergyTransferStarted*

Main (Test scenario)

| | |
|--|------|
| Charging Station | CSMS |
| 1. Execute Reusable State <i>ParkingBayUnoccupied</i> | |
| <p><u>Note(s):</u></p> <ul style="list-style-type: none"> - This step will be executed after the <i><Configured transaction duration></i> is reached. - This causes the transaction to stop. | |

Tool validations

| |
|---|
| N/a |
| <p>Post scenario validations:</p> <ul style="list-style-type: none"> - The TransactionEventRequest containing eventType <i>Ended</i> contains the MeterValue field. - The MeterValue must contain <i><An element per data collection moment indicated by SampledDataTxEndedInterval. The Test System will not validate this.></i> - timestamp <i><The intervals between the timestamps of the received Meter Value messages must equal the configured value at SampledDataTxEndedInterval.></i> - sampledValue[0].context must be <i>Sample.Periodic</i> AND one must have <i>Transaction.End</i> - sampledValue must contain <i><An element per configured measurand at the SampledDataTxEndedMeasurands. The measurand field may be omitted when the measurand is "Energy.Active.Import.Register"></i> - sampledValue.signedMeterValue should not be omitted - sampledValue.signedMeterValue.publicKey should exist and depending on the value of OCPPCommCtrlr.PublicKeyWithSignedMeterValue, should be either "", or a valid public key - None of the provided sampledValues shall have location = <i>EV</i>, except when measurand = <i>SoC</i>. |

K Smart Charging

Determine Charging Profile Limit Multiplier

Not all chargers support setting limits in A or W. This can be configured with the configuration variable *<Configured chargingRateUnit>*. To calculate the limit to be used, the following rules must be followed:

If *<Configured chargingRateUnit>* is A, then *<limit multiplier>* is 1

If *<Configured chargingRateUnit>* is W and *<Configured numberPhases>* is 1, then *<limit multiplier>* is 230

If *<Configured chargingRateUnit>* is W and *<Configured numberPhases>* is 2, then *<limit multiplier>* is 460

If *<Configured chargingRateUnit>* is W and *<Configured numberPhases>* is 3, then *<limit multiplier>* is 690

Example 1

Given a test case is configured with:

```
<Configured chargingRateUnit> W
<Configured numberPhases> 2
```

When the test case specifies:

```
chargingProfile.chargingSchedule.chargingSchedulePeriod.limit 6 * <limit multiplier>
```

Then it should set

```
chargingProfile.chargingSchedule.chargingSchedulePeriod.limit 2760
```

Example 2

Given a test case is configured with:

```
<Configured chargingRateUnit> A
<Configured numberPhases> 3
```

When the test case specifies:

```
chargingProfile.chargingSchedule.chargingSchedulePeriod.limit 6 * <limit multiplier>
```

Then it should set

```
chargingProfile.chargingSchedule.chargingSchedulePeriod.limit 6
```

TC_K_01_CS: Set Charging Profile - TxDefaultProfile - Specific EVSE

| Test case name | Set Charging Profile - TxDefaultProfile - Specific EVSE |
|-------------------|--|
| Test case Id | TC_K_01_CS |
| Use case Id(s) | K01 |
| Requirement(s) | K01.FR.07, K01.FR.15 |
| System under test | Charging Station |
| Description | To enable the CSMS to influence the charging power or current drawn from a specific EVSE or the entire Charging Station over a period of time. The CSMS sends a SetChargingProfileRequest to the Charging Station to influence the power or current drawn by EVs. The CSMS calculates a ChargingSchedule to stay within certain limits, which MAY be imposed by any external system. |
| Purpose | To verify if the Charging station is able to accept and successfully change to the TxDefaultProfile charging profile sent by the CSMS on a specific EVSE as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a SetChargingProfileResponse | 1. The Test System sends a SetChargingProfileRequest with evseld <Configured evseld> AND chargingProfile.id <Configured chargingProfileId> AND chargingProfile.chargingProfilePurpose TxDefaultProfile chargingProfile.chargingSchedule.duration <Configured duration> chargingProfile.chargingSchedule.chargingRateUnit <Configured chargingRateUnit> chargingProfile.chargingSchedule.chargingSchedulePeriod.startPeriod 0 chargingProfile.chargingSchedule.chargingSchedulePeriod.limit 6 * <limit multiplier> Note: Check Determine Charging Profile Limit Multiplier for <limit multiplier> chargingProfile.chargingSchedule.chargingSchedulePeriod.numberPhases <Configured numberPhases> |
| 4. The Charging Station responds with a GetChargingProfilesResponse | 3. The Test System sends a GetChargingProfilesRequest with chargingProfile.chargingProfileId <Configured chargingProfileId> requestId <Generated requestId> |
| 5. The Charging Station sends a ReportChargingProfilesRequest | 6. The Test System responds with a ReportChargingProfilesResponse |
| Note(s): - If tbc is True at Step 5 then step 5 and 6 will be repeated | |

| |
|---|
| Tool validations |
| * Step 2: Message SetChargingProfileResponse - status Accepted * Step 4: Message GetChargingProfilesResponse - status Accepted * Step 5: Message ReportChargingProfilesRequest - requestId <Generated requestId> - evseld <Configured EVSEId>* - chargingProfile <Configured ChargingProfile> |
| Post scenario validations: - The same profile is reported as send in step 1 |

TC_K_02_CS: Set Charging Profile - TxProfile without ongoing transaction on the specified EVSE

| | |
|--------------------------|--|
| Test case name | Set Charging Profile - TxProfile without ongoing transaction on the specified EVSE |
| Test case Id | TC_K_02_CS |
| Use case Id(s) | K01 |
| Requirement(s) | K01.FR.04, K01.FR.07, K01.FR.09 |
| System under test | Charging Station |
| Description | To enable the CSMS to influence the charging power or current drawn from a specific EVSE or the entire Charging Station over a period of time. The CSMS sends a SetChargingProfileRequest to the Charging Station to influence the power or current drawn by EVs. The CSMS calculates a ChargingSchedule to stay within certain limits, which MAY be imposed by any external system. |
| Purpose | To verify if the Charging station is able to accept and successfully change to the a TxProfile charging profile, without ongoing transaction, sent by the CSMS on a specific EVSE as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a SetChargingProfileResponse | 1. The Test System sends a SetChargingProfileRequest with evseld <Configured evseld> AND chargingProfile.id <Configured chargingProfileId> AND chargingProfile.chargingProfilePurpose TxProfile AND chargingProfile.transactionId UNKNOWN-TRANSACTION-ID |

| |
|---|
| Tool validations |
| * Step 2: Message SetChargingProfileResponse - status <i>Rejected</i> |
| Post scenario validations: - N/a |

TC_K_03_CS: Set Charging Profile - ChargingStationMaxProfile

| | |
|-------------------|--|
| Test case name | Set Charging Profile - ChargingStationMaxProfile |
| Test case Id | TC_K_03_CS |
| Use case Id(s) | K01 |
| Requirement(s) | N/a |
| System under test | Charging Station |
| Description | To enable the CSMS to influence the charging power or current drawn from a specific EVSE or the entire Charging Station over a period of time. The CSMS sends a SetChargingProfileRequest to the Charging Station to influence the power or current drawn by EVs. The CSMS calculates a ChargingSchedule to stay within certain limits, which MAY be imposed by any external system. |
| Purpose | To verify if the Charging station is able to accept and successfully change to the ChargingStationMaxProfile sent by the CSMS on a specific EVSE as described at the OCPP specification. |
| Prerequisite(s) | n/a |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State:

N/a

Main (Test scenario)

| Charging Station | CSMS |
|---|---|
| 2. The Charging Station responds with a SetChargingProfileResponse | 1. The Test System sends a SetChargingProfileRequest with chargingProfile.id <Configured chargingProfileId> AND chargingProfile.chargingProfilePurpose <i>ChargingStationMaxProfile</i> AND chargingProfile.chargingProfileKind <i>Absolute</i> AND chargingProfile.chargingSchedule.duration <Configured duration> AND chargingProfile.chargingSchedule.chargingRateUnit <Configured chargingRateUnit> AND chargingProfile.chargingSchedule.chargingSchedulePeriod.startPeriod 0 AND chargingProfile.chargingSchedule.chargingSchedulePeriod.limit 6 * <limit multiplier> AND <i>Note: Check Determine Charging Profile Limit Multiplier for <limit multiplier></i> chargingProfile.chargingSchedule.chargingSchedulePeriod.numberPhases <Configured numberPhases> AND EVSEId 0 |
| 4. The Charging Station responds with a GetChargingProfilesResponse | 3. The Test System sends a GetChargingProfilesRequest with chargingProfile.chargingProfileId <Configured chargingProfileId> requestId <Generated requestId> |
| 5. The Charging Station sends a ReportChargingProfilesRequest | 6. The Test System responds with a ReportChargingProfilesResponse |
| Note(s): - If tbc is True at Step 5 then step 5 and 6 will be repeated | |

| Tool validations |
|---|
| <div>* Step 2: Message SetChargingProfileResponse - status <i>Accepted</i></div> <div>* Step 4: Message GetChargingProfilesResponse - status <i>Accepted</i></div> <div>* Step 5: Message ReportChargingProfilesRequest - requestId <i><Generated requestId></i> - EvseId <i>0</i> - chargingProfile <i><Generated chargingProfile></i></div> |
| <div>Post scenario validations: - The same profile is reported as send in step 1</div> |

TC_K_04_CS: Replace charging profile - With chargingProfileId

| | |
|-------------------|--|
| Test case name | Replace charging profile - With chargingProfileId |
| Test case Id | TC_K_04_CS |
| Use case Id(s) | K01 |
| Requirement(s) | K01.FR.05 |
| System under test | Charging Station |
| Description | To enable the CSMS to influence the charging power or current drawn from a specific EVSE or the entire Charging Station over a period of time. The CSMS sends a SetChargingProfileRequest to the Charging Station to influence the power or current drawn by EVs. The CSMS calculates a ChargingSchedule to stay within certain limits, which MAY be imposed by any external system. |
| Purpose | To verify if the Charging station is able to accept and successfully change to the ChargingStationMaxProfile sent by the CSMS on a specific EVSE as described at the OCPP specification. |
| Prerequisite(s) | n/a |

Before (Preparations)

Configuration State:

N/a

Memory State:

A chargeprofile with <Generated chargingProfileId> AND limit 6.0/6000.0 AND ChargingProfilePurpose TxDefaultProfile is configured

Reusable State:

N/a

Main (Test scenario)

| Charging Station | CSMS |
|--|---|
| 2. The Charging Station responds with a SetChargingProfileResponse | 1. The Test System sends a SetChargingProfileRequest with chargingProfile.id <Configured chargingProfileId> chargingProfile.chargingProfilePurpose TxDefaultProfile chargingProfile.chargingSchedule.chargingSchedulePeriod.limit 10 * <limit multiplier> Note: Check Determine Charging Profile Limit Multiplier for <limit multiplier> |
| 4. The Charging Station responds with a GetChargingProfilesResponse | 3. The Test System sends a GetChargingProfilesRequest with chargingProfile.chargingProfileId <Configured chargingProfileId> |
| 5. The Charging Station sends a ReportChargingProfilesRequest | 6. The Test System responds with a ReportChargingProfilesResponse |

Note(s):

- If **tb** is True at Step 5 then step 5 and 6 will be repeated

Tool validations

* Step 2:

Message **SetChargingProfileResponse**

- **status** Accepted

* Step 4:

Message **GetChargingProfilesResponse**

- **status** Accepted

* Step 5:

Message **ReportChargingProfilesRequest**

- **requestId** Same Id as in the GetChargingProfilesRequest in step 3

- **EVSEId** <Configured EVSEId>

- **chargingProfile** <ChargingProfile set in step 1>

| |
|--|
| Tool validations |
| Post scenario validations: - N/a |

TC_K_05_CS: Clear Charging Profile - With chargingProfileId

| | |
|--------------------------|--|
| Test case name | Clear Charging Profile - With chargingProfileId |
| Test case Id | TC_K_05_CS |
| Use case Id(s) | K10 |
| Requirement(s) | K10.FR.03 |
| System under test | Charging Station |
| Description | If the CSMS wishes to clear some or all of the charging profiles that were previously sent to the Charging Station, then the CSMS sends a ClearChargingProfileRequest to the Charging Station. |
| Purpose | To verify if the Charging station is able to accept the request and clear a specific charging profile sent with only a chargingProfileId by the CSMS as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: A chargingprofile with <Configured chargingProfileId> is configured |
| Reusable State: N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a ClearChargingProfileResponse | 1. The Test System sends a ClearChargingProfileRequest with chargingProfileId <Configured chargingProfileId> |
| 4. The Charging Station responds with a GetChargingProfilesResponse | 3. The Test System sends a GetChargingProfilesRequest with chargingProfile.chargingProfileId <Configured chargingProfileId> |

| |
|---|
| Tool validations |
| * Step 2: Message ClearChargingProfileResponse - status <i>Accepted</i> * Step 4: Message GetChargingProfilesResponse - status <i>NoProfiles</i> |
| Post scenario validations: - N/a |

TC_K_06_CS: Clear Charging Profile - With stackLevel/purpose combination for one profile

| | |
|-------------------|--|
| Test case name | Clear Charging Profile - With stackLevel/purpose combination for one profile |
| Test case Id | TC_K_06_CS |
| Use case Id(s) | K10 |
| Requirement(s) | K10.FR.04 |
| System under test | Charging Station |
| Description | If the CSMS wishes to clear some or all of the charging profiles that were previously sent to the Charging Station, then the CSMS sends a ClearChargingProfileRequest to the Charging Station. |
| Purpose | To verify if the Charging station is able to accept the request and clear a charging profile sent with a stackLevel/purpose combination by the CSMS as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: A chargingprofile with <Configured chargingProfilePurpose> AND <Configured stackLevel> is configured |
| Reusable State: EnergyTransferStarted |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a ClearChargingProfileResponse | 1. The Test System sends a ClearChargingProfileRequest with chargingProfileCriteria.chargingProfilePurpose <Configured chargingProfilePurpose> AND chargingProfileCriteria.stackLevel <Configured stackLevel> |
| 4. The Charging Station responds with a GetChargingProfilesResponse | 3. The Test System sends a GetChargingProfilesRequest with chargingProfile.chargingProfilePurpose <Configured chargingProfilePurpose> AND chargingProfile.stackLevel <Configured stackLevel> |

| |
|--|
| Tool validations |
| * Step 2: Message ClearChargingProfileResponse - status <i>Accepted</i> |
| * Step 4: Message GetChargingProfilesResponse - status <i>NoProfiles</i> |
| Post scenario validations: - N/a |

TC_K_07_CS: Clear Charging Profile - With unknown stackLevel/purpose combination

| | |
|--------------------------|---|
| Test case name | Clear Charging Profile - With unknown stackLevel/purpose combination |
| Test case Id | TC_K_07_CS |
| Use case Id(s) | K10 |
| Requirement(s) | K10.FR.01 |
| System under test | Charging Station |
| Description | If the CSMS wishes to clear some or all of the charging profiles that were previously sent to the Charging Station, then the CSMS sends a ClearChargingProfileRequest to the Charging Station. |
| Purpose | To verify if the Charging station is able to deny the request to clear a specific charging profile when an unknown chargingProfileId and unknown stackLevel/purpose combination is sent by the CSMS as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: A chargingprofile with ChargingProfilePurpose TxDefaultProfile AND StackLevel 1 is configured |
| Reusable State: N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a ClearChargingProfileResponse | 1. The Test System sends a ClearChargingProfileRequest with chargingProfileCriteria.chargingProfilePurpose <i>ChargingStationMaxProfile</i> AND chargingProfileCriteria.stackLevel 0 |

| |
|--|
| Tool validations |
| * Step 2: Message ClearChargingProfileResponse - status <i>Unknown</i> |
| Post scenario validations: - N/a |

TC_K_08_CS: Clear Charging Profile - Without previous charging profile

| | |
|-------------------|---|
| Test case name | Clear Charging Profile - Without previous charging profile |
| Test case Id | TC_K_08_CS |
| Use case Id(s) | K10 |
| Requirement(s) | K10.FR.01 |
| System under test | Charging Station |
| Description | If the CSMS wishes to clear some or all of the charging profiles that were previously sent to the Charging Station, then the CSMS sends a ClearChargingProfileRequest to the Charging Station. |
| Purpose | To verify if the Charging station is able to deny the request to clear a specific charging profile when no charging profiles are configured as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a ClearChargingProfileResponse | 1. The Test System sends a ClearChargingProfileRequest with chargingProfileId <Generated <i>chargingProfileId</i> > |

| |
|--|
| Tool validations |
| * Step 2: Message ClearChargingProfileResponse - status <i>Unknown</i> |
| Post scenario validations: - N/a |

TC_K_09_CS: Clear Charging Profile - Clearing a TxDefaultProfile - With ongoing transaction

| | |
|-------------------|--|
| Test case name | Clear Charging Profile - Clearing a TxDefaultProfile - With ongoing transaction |
| Test case Id | TC_K_09_CS |
| Use case Id(s) | K10 |
| Requirement(s) | K10.FR.07 |
| System under test | Charging Station |
| Description | If the CSMS wishes to clear some or all of the charging profiles that were previously sent to the Charging Station, then the CSMS sends a ClearChargingProfileRequest to the Charging Station. |
| Purpose | To verify if the Charging station is able to accept the request and clear a TxDefaultProfile by the CSMS as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|---|
| Before (Preparations) |
| Configuration State: SmartChargingCtrlr.LimitChangeSignificance is <i>1.0</i> |
| Memory State: SetChargingProfile with ChargingProfile 1: chargingProfilePurpose is <i>TxDefaultProfile</i> chargingProfileKind should be <i>Absolute</i> stackLevel should be <i>0</i> evseld <Configured evseld> startSchedule <current dateTime> numberPhases <Configured numberPhases> ChargingSchedule: duration 400 + <Configured max time deviation> chargingRateUnit <Configured chargingRateUnit> startPeriod 0, limit 6 * <limit multiplier> Note: Check Determine Charging Profile Limit Multiplier for <limit multiplier> |
| Reusable State: State is EnergyTransferStarted |

| Main (Test scenario) | |
|---|---|
| Charging Station | CSMS |
| 2. The Charging Station responds with a GetCompositeScheduleResponse | 1. The Test System sends a GetCompositeScheduleRequest with evseld is <Configured evseld> |
| 4. The Charging Station responds with a ClearChargingProfileResponse | 3. The Test System sends a ClearChargingProfileRequest with chargingProfileCriteria.chargingProfilePurpose TxDefaultProfile |
| 5. The Charging Station responds with a GetCompositeScheduleResponse | 6. The Test System sends a GetCompositeScheduleRequest with evseld <Configured evseld> duration is 300 chargingRateUnit <Configured chargingRateUnit> |

| Tool validations |
|---|
| <p>* Step 2:</p> <p>(Message: GetCompositeScheduleResponse)</p> <p>status <i>Accepted</i></p> <p>evseld <i><Configured evseld></i></p> <p>ChargingSchedule:</p> <p>duration <i>300</i></p> <p>chargingRateUnit <i><Configured chargingRateUnit></i></p> <p>scheduleStart <i><The time the GetCompositeScheduleRequest was transmitted +/- <Configured max time deviation>></i></p> <p>startPeriod <i>0, limit</i> <i>6 * <limit multiplier></i></p> <p>Note: Check Determine Charging Profile Limit Multiplier for <i><limit multiplier></i></p> <p>* Step 4:</p> <p>(Message: ClearChargingProfileResponse)</p> <p>status <i>is Accepted</i></p> <p>* Step 5:</p> <p>(Message: GetCompositeScheduleResponse)</p> <p>status <i>Accepted</i></p> <p>evseld <i><Configured evseld></i></p> <p>ChargingSchedule:</p> <p>duration <i>300</i></p> <p>chargingRateUnit <i><Configured chargingRateUnit></i></p> <p>scheduleStart <i><The time the GetCompositeScheduleRequest was transmitted +/- <Configured max time deviation>></i></p> <p>startPeriod <i>0, limit</i> <i><Local limit of Charging Station (> 6 * <limit multiplier>)></i></p> <p>Note: Check Determine Charging Profile Limit Multiplier for <i><limit multiplier></i></p> |
| <p>Post scenario validations:</p> <p>N/a</p> |

TC_K_10_CS: Set Charging Profile - TxDefaultProfile - All EVSE

| | |
|-------------------|--|
| Test case name | Set Charging Profile - TxDefaultProfile - All EVSE |
| Test case Id | TC_K_10_CS |
| Use case Id(s) | K01 |
| Requirement(s) | K01.FR.07, K01.FR.14 |
| System under test | Charging Station |
| Description | To enable the CSMS to influence the charging power or current drawn from a specific EVSE or the entire Charging Station over a period of time. The CSMS sends a SetChargingProfileRequest to the Charging Station to influence the power or current drawn by EVs. The CSMS calculates a ChargingSchedule to stay within certain limits, which MAY be imposed by any external system. |
| Purpose | To verify if the Charging station is able to accept and successfully change to the TxDefaultProfile charging profile sent by the CSMS for all EVSE as described at the OCPP specification. |
| Prerequisite(s) | n/a |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State:

N/a

Main (Test scenario)

| Charging Station | CSMS |
|---|---|
| 2. The Charging Station responds with a SetChargingProfileResponse | <p>1. The Test System sends a SetChargingProfileRequest with evseld 0 AND chargingProfile.id <Configured chargingProfileId> AND chargingProfile.chargingProfilePurpose TxDefaultProfile chargingProfile.chargingSchedule.duration <Configured duration> chargingProfile.chargingSchedule.chargingRateUnit <Configured chargingRateUnit> chargingProfile.chargingSchedule.chargingSchedulePeriod.startPeriod 0 chargingProfile.chargingSchedule.chargingSchedulePeriod.limit 6.0 * <limit multiplier> Note: Check Determine Charging Profile Limit Multiplier for <limit multiplier> chargingProfile.chargingSchedule.chargingSchedulePeriod.numberPhases <Configured numberPhases></p> |
| 4. The Charging Station responds with a GetChargingProfilesResponse | 3. The Test System sends a GetChargingProfilesRequest with chargingProfile.chargingProfileId <Configured chargingProfileId> requestId <Generated requestId> |
| 5. The Charging Station sends a ReportChargingProfilesRequest | 6. The Test System responds with a ReportChargingProfilesResponse |
| <u>Note(s):</u> - If tbc is True at Step 5 then step 5 and 6 will be repeated | |

| Tool validations |
|---|
| <p>* Step 2:</p> <p>Message SetChargingProfileResponse</p> <ul style="list-style-type: none">- status <i>Accepted</i> <p>* Step 4:</p> <p>Message GetChargingProfilesResponse</p> <ul style="list-style-type: none">- status <i>Accepted</i> <p>* Step 5:</p> <p>Message ReportChargingProfilesRequest</p> <ul style="list-style-type: none">- requestId <i><Generated requestId></i>- EVSEId <i>0</i>- tbc <i>false</i>- chargingProfile <i><Configured chargingProfile></i> |
| <p>Post scenario validations:</p> <ul style="list-style-type: none">- The same profile is reported as send in step 1 |

TC_K_11_CS: Set Charging Profile - Unable to set TxProfile on all EVSE at once

| | |
|-------------------|--|
| Test case name | Set Charging Profile - Unable to set TxProfile on all EVSE at once |
| Test case Id | TC_K_11_CS |
| Use case Id(s) | K01 |
| Requirement(s) | K01.FR.04, K01.FR.16 |
| System under test | Charging Station |
| Description | To enable the CSMS to influence the charging power or current drawn from a specific EVSE or the entire Charging Station over a period of time. The CSMS sends a SetChargingProfileRequest to the Charging Station to influence the power or current drawn by EVs. The CSMS calculates a ChargingSchedule to stay within certain limits, which MAY be imposed by any external system. |
| Purpose | To verify if the Charging station is able to deny a TxProfile when sent to all EVSE as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a SetChargingProfileResponse | 1. The Test System sends a SetChargingProfileRequest with evseld 0 AND chargingProfile.id <Configured chargingProfileId> AND chargingProfile.chargingProfilePurpose TxProfile |

| |
|--|
| Tool validations |
| * Step 2: Message SetChargingProfileResponse - status Rejected |
| Post scenario validations: - N/a |

TC_K_12_CS: Set Charging Profile - ChargerRateUnit Rejected

| | |
|-------------------|--|
| Test case name | Set Charging Profile - ChargerRateUnit Rejected |
| Test case Id | TC_K_12_CS |
| Use case Id(s) | K01 |
| Requirement(s) | K01.FR.26 |
| System under test | Charging Station |
| Description | To enable the CSMS to influence the charging power or current drawn from a specific EVSE or the entire Charging Station over a period of time. The CSMS sends a SetChargingProfileRequest to the Charging Station to influence the power or current drawn by EVs. The CSMS calculates a ChargingSchedule to stay within certain limits, which MAY be imposed by any external system. |
| Purpose | To verify if the Charging station is able to deny a chargeProfile when the given ChargerRateUnit is not known by the charger as described at the OCPP specification. |
| Prerequisite(s) | |

Before (Preparations)

Configuration State:

This testcase can only be tested when one of the 2 chargingRateUnits is not supported.

Memory State:

N/a

Reusable State:

N/a

Main (Test scenario)

| Charging Station | CSMS |
|--|---|
| 2. The Charging Station responds with a SetChargingProfileResponse | 1. The Test System sends a SetChargingProfileRequest with chargingProfile.id <Configured chargingProfileId> AND chargingProfile.chargingSchedule.chargingRateUnit <Configured chargingRateUnit> |

Tool validations

* Step 2:

Message SetChargingProfileResponse

- **status** *Rejected*

Post scenario validations:

- N/a

TC_K_13_CS: Set Charging Profile - Persistent over reboot

| | |
|-------------------|--|
| Test case name | Set Charging Profile - Persistent over reboot |
| Test case Id | TC_K_13_CS |
| Use case Id(s) | K01 |
| Requirement(s) | K01.FR.27 |
| System under test | Charging Station |
| Description | To enable the CSMS to influence the charging power or current drawn from a specific EVSE or the entire Charging Station over a period of time. The CSMS sends a SetChargingProfileRequest to the Charging Station to influence the power or current drawn by EVs. The CSMS calculates a ChargingSchedule to stay within certain limits, which MAY be imposed by any external system. |
| Purpose | To verify if the Charging station is able to save a chargingProfile persistent over reboot as described at the OCPP specification. |
| Prerequisite(s) | n/a |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State:

N/a

Main (Test scenario)

| Charging Station | CSMS |
|---|---|
| 2. The Charging Station responds with a SetChargingProfileResponse | <p>1. The Test System sends a SetChargingProfileRequest with evseld <Configured evseld> AND chargingProfile.id <Configured chargingProfileId> AND chargingProfile.chargingProfilePurpose TxDefaultProfile chargingProfile.chargingSchedule.duration <Configured duration> chargingProfile.chargingSchedule.chargingRateUnit <Configured chargingRateUnit> chargingProfile.chargingSchedule.chargingSchedulePeriod.startPeriod 0 chargingProfile.chargingSchedule.chargingSchedulePeriod.limit 6 * <limit multiplier> Note: Check Determine Charging Profile Limit Multiplier for <limit multiplier> chargingProfile.chargingSchedule.chargingSchedulePeriod.numberPhases <Configured numberPhases></p> |
| 3. Execute Reusable State <i>Booted</i> | |
| 5. The Charging Station responds with a GetChargingProfilesResponse | 4. The Test System sends a GetChargingProfilesRequest with chargingProfile.chargingProfileId <Configured chargingProfileId> |
| 6. The Charging Station sends a ReportChargingProfilesRequest | 7. The Test System responds with a ReportChargingProfilesResponse |
| Note(s): - If tbc is True at Step 6 then step 6 and 7 will be repeated | |

| Tool validations |
|--|
| <p>* Step 2:</p> <p>Message SetChargingProfileResponse</p> <ul style="list-style-type: none">- status <i>Accepted</i> <p>* Step 5:</p> <p>Message GetChargingProfilesResponse</p> <ul style="list-style-type: none">- status <i>Accepted</i> <p>* Step 6:</p> <p>Message ReportChargingProfilesRequest</p> <ul style="list-style-type: none">- requestId <i>Same Id as in the GetChargingProfilesRequest in step 4</i>- EVSEId <i><Configured EVSEId></i>- chargingProfile <i><Configured chargingProfile></i> |
| <p>Post scenario validations:</p> <ul style="list-style-type: none">- The same profile is reported as send in step 1 |

TC_K_14_CS: Set Charging Profile - Unexisting EVSEid

| | |
|-------------------|--|
| Test case name | Set Charging Profile - Unexisting EVSEid |
| Test case Id | TC_K_14_CS |
| Use case Id(s) | K01 |
| Requirement(s) | K01.FR.28 |
| System under test | Charging Station |
| Description | To enable the CSMS to influence the charging power or current drawn from a specific EVSE or the entire Charging Station over a period of time. The CSMS sends a SetChargingProfileRequest to the Charging Station to influence the power or current drawn by EVs. The CSMS calculates a ChargingSchedule to stay within certain limits, which MAY be imposed by any external system. |
| Purpose | To verify if the Charging station is able to reject a chargingProfile when the provided EVSEid is unknown as described at the OCPP specification. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a SetChargingProfileResponse | 1. The Test System sends a SetChargingProfileRequest with evseld <EVSECount + 1> |

| |
|---|
| Tool validations |
| * Step 2: Message SetChargingProfileResponse - status <i>Rejected</i> |
| Post scenario validations: - N/a |

TC_K_15_CS: Set Charging Profile - Not Supported

| | |
|--------------------------|--|
| Test case name | Set Charging Profile - Not Supported |
| Test case Id | TC_K_15_CS |
| Use case Id(s) | K01 |
| Requirement(s) | K01.FR.29 |
| System under test | Charging Station |
| Description | To enable the CSMS to influence the charging power or current drawn from a specific EVSE or the entire Charging Station over a period of time. The CSMS sends a SetChargingProfileRequest to the Charging Station to influence the power or current drawn by EVs. The CSMS calculates a ChargingSchedule to stay within certain limits, which MAY be imposed by any external system. |
| Purpose | To verify if the Charging station is able to raise a callerror when it does not support smart charging as described at the OCPP specification. |
| Prerequisite(s) | Charging station does not support smart charging |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with RPC Framework: CALLERROR: NotSupported. | 1. The Test System sends a SetChargingProfileRequest with chargingProfile.id <Configured chargingProfileId> |

| |
|--|
| Tool validations |
| - N/a |
| Post scenario validations: - N/a |

TC_K_16_CS: Set Charging Profile - Unknown transactionId

| | |
|-------------------|--|
| Test case name | Set Charging Profile - Unknown transactionId |
| Test case Id | TC_K_16_CS |
| Use case Id(s) | K01 |
| Requirement(s) | K01.FR.33 |
| System under test | Charging Station |
| Description | To enable the CSMS to influence the charging power or current drawn from a specific EVSE or the entire Charging Station over a period of time. The CSMS sends a SetChargingProfileRequest to the Charging Station to influence the power or current drawn by EVs. The CSMS calculates a ChargingSchedule to stay within certain limits, which MAY be imposed by any external system. |
| Purpose | To verify if the Charging station is able to reject a charge profile when an unknown transactionId is provided as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: State is <i>EnergyTransferStarted</i> |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a SetChargingProfileResponse | 1. The Test System sends a SetChargingProfileRequest with evseld <Configured evseld> AND chargingProfile.id <Configured chargingProfileId> AND chargingProfile.chargingProfilePurpose TxProfile AND chargingProfile.transactionId UNKNOWN-TRANSACTION-ID |

| |
|--|
| Tool validations |
| * Step 2: Message SetChargingProfileResponse - status Rejected |
| Post scenario validations: - N/a |

TC_K_19_CS: Set Charging Profile - ChargingProfileKind is Recurring

| | |
|-------------------|--|
| Test case name | Set Charging Profile - ChargingProfileKind is Recurring |
| Test case Id | TC_K_19_CS |
| Use case Id(s) | K01 |
| Requirement(s) | K01.FR.40 |
| System under test | Charging Station |
| Description | To enable the CSMS to influence the charging power or current drawn from a specific EVSE or the entire Charging Station over a period of time. The CSMS sends a SetChargingProfileRequest to the Charging Station to influence the power or current drawn by EVs. The CSMS calculates a ChargingSchedule to stay within certain limits, which MAY be imposed by any external system. |
| Purpose | To verify if the Charging station is able to accept and successfully change to the Recurring ChargingProfileKind sent by the CSMS as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a SetChargingProfileResponse | 1. The Test System sends a SetChargingProfileRequest with chargingProfile.chargingProfileKind Recurring chargingProfile.recurrencyKind <Configured RecurrencyKind> |

| |
|--|
| Tool validations |
| * Step 2: Message SetChargingProfileResponse - status Accepted |
| Post scenario validations: - N/a |

TC_K_21_CS: Set Charging Profile - ValidFrom

| | |
|-------------------|---|
| Test case name | Set Charging Profile - ValidFrom |
| Test case Id | TC_K_21_CS |
| Use case Id(s) | K01 |
| Requirement(s) | K01.FR.36 |
| System under test | Charging Station |
| Description | The CSMS is able to impose charging limits by setting a charging profile that becomes valid after a certain date/time using the SetChargingProfileRequest message. It is only tested on EVSE #1, because mechanism is the same regardless of EVSE. |
| Purpose | To verify if the Charging Station activates a set charging profile after the ValidFrom is reached. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a SetChargingProfileResponse | <p>1. The Test System sends a SetChargingProfileRequest with chargingProfile.chargingProfilePurpose is <i>TxDefaultProfile</i> chargingProfile.chargingProfileKind is <i>Relative</i> evseld <configured evseld> chargingProfile.validFrom <current dateTime + 300 seconds> chargingProfile.validTo is absent chargingProfile.chargingSchedule[0].startSchedule is absent chargingProfile.chargingSchedule[0].chargingSchedulePeriod[0].numberPhases <Configured numberPhases> chargingProfile.chargingSchedule[0].chargingSchedulePeriod[0].startPeriod 0 chargingProfile.chargingSchedule[0].chargingSchedulePeriod[0].limit 6 * <limit multiplier> Note: Check Determine Charging Profile Limit Multiplier for <limit multiplier></p> |
| 4. The Charging Station responds with a GetCompositeScheduleResponse | <p>3. The Test System sends a GetCompositeScheduleRequest with evseld <configured evseld> duration is 400 chargingRateUnit <Configured chargingRateUnit></p> |

| Tool validations |
|---|
| <p>* Step 2: (Message: SetChargingProfileResponse) status is <i>Accepted</i></p> <p>* Step 4: (Message: GetCompositeScheduleResponse) status <i>Accepted</i> schedule.evseId <i><configured evseId></i> schedule.chargingRateUnit <i><Configured chargingRateUnit></i> schedule.duration <i>400</i> schedule.scheduleStart <i><The time the GetCompositeScheduleRequest was transmitted +/- <Configured max time deviation>></i> schedule.chargingSchedulePeriod[0].startPeriod <i>0</i> schedule.chargingSchedulePeriod[0].limit <i><Local limit of Charging Station (> 6)></i> schedule.chargingSchedulePeriod[1].startPeriod <i>(300 - x)</i> schedule.chargingSchedulePeriod[1].limit <i>6.0 * <limit multiplier></i></p> <p>Note: Check Determine Charging Profile Limit Multiplier for <i><limit multiplier></i> Note: The period of time between the <i>scheduleStart</i> from the <i>SetChargingProfileRequest</i> and the <i>scheduleStart</i> from the <i>GetCompositeScheduleResponse</i> is called <i>x</i>.</p> |
| <p>Post scenario validations: N/a</p> |

TC_K_22_CS: Set Charging Profile - ValidTo

| | |
|-------------------|---|
| Test case name | Set Charging Profile - ValidTo |
| Test case Id | TC_K_22_CS |
| Use case Id(s) | K01 |
| Requirement(s) | K01.FR.37 |
| System under test | Charging Station |
| Description | The CSMS is able to impose charging limits by setting a charging profile using the SetChargingProfileRequest message. |
| Purpose | To verify if the Charging Station deactivates a set charging profile after the ValidTo has passed. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a SetChargingProfileResponse | <p>1. The Test System sends a SetChargingProfileRequest with chargingProfile.chargingProfilePurpose is <i>TxDefaultProfile</i> chargingProfile.chargingProfileKind is <i>Absolute</i> evseld <Configured evseld> chargingProfile.validFrom <current dateTime - <Configured max time deviation> seconds> chargingProfile.validTo <current dateTime + 300 seconds> chargingProfile.chargingSchedule[0].startSchedule <current dateTime - <Configured max time deviation> seconds> chargingProfile.chargingSchedule[0].chargingSchedulePeriod[0].numberPhases <Configured numberPhases> chargingProfile.chargingSchedule[0].chargingSchedulePeriod[0].startPeriod 0 chargingProfile.chargingSchedule[0].chargingSchedulePeriod[0].limit 6 * <limit multiplier> <i>Note: Check Determine Charging Profile Limit Multiplier for <limit multiplier></i></p> |
| 4. The Charging Station responds with a GetCompositeScheduleResponse | <p>3. The Test System sends a GetCompositeScheduleRequest with evseld 1 duration is 400 chargingRateUnit <Configured chargingRateUnit></p> |

| Tool validations |
|---|
| <p>* Step 2: (Message: SetChargingProfileResponse) status is <i>Accepted</i></p> <p>* Step 4: (Message: GetCompositeScheduleResponse) status <i>Accepted</i> evseld <Configured evseld> chargingRateUnit <Configured chargingRateUnit> ChargingSchedule: duration 400 chargingRateUnit <Configured chargingRateUnit> scheduleStart <The time the GetCompositeScheduleRequest was transmitted +/- <Configured max time deviation>> <i>Note: The period of time between the <code>scheduleStart</code> from the SetChargingProfileRequest and the <code>scheduleStart</code> from the GetCompositeScheduleResponse is called x.</i> startPeriod 0, limit 6 * <limit multiplier> startPeriod (300 - x), limit <Local limit of Charging Station (> 6 * <limit multiplier>)> <i>Note: Check Determine Charging Profile Limit Multiplier for <limit multiplier></i></p> |
| <p>Post scenario validations: N/a</p> |

TC_K_23_CS: Set Charging Profile - StartSchedule

| | |
|-------------------|---|
| Test case name | Set Charging Profile - StartSchedule |
| Test case Id | TC_K_23_CS |
| Use case Id(s) | K01 |
| Requirement(s) | K01.FR.30 |
| System under test | Charging Station |
| Description | The CSMS is able to impose charging limits by setting a charging profile using the SetChargingProfileRequest message. |
| Purpose | To verify if the Charging Station activates a set charging profile after the StartSchedule has passed. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a SetChargingProfileResponse | <p>1. The Test System sends a SetChargingProfileRequest with chargingProfile.chargingProfilePurpose is <i>TxDefaultProfile</i> chargingProfile.chargingProfileKind is <i>Absolute</i> evseld <configured evseld> chargingProfile.chargingSchedule[0].startSchedule <current dateTime> + 60 seconds> chargingProfile.chargingSchedule[0].chargingSchedulePeriod[].numberPhases <Configured numberPhases> chargingProfile.chargingSchedule[0].chargingSchedulePeriod[].startPeriod 0 chargingProfile.chargingSchedule[0].chargingSchedulePeriod[0].limit 6 * <limit multiplier> Note: Check Determine Charging Profile Limit Multiplier for <limit multiplier></p> |
| 4. The Charging Station responds with a GetCompositeScheduleResponse | <p>3. The Test System sends a GetCompositeScheduleRequest with evseld <Configured evseld> duration is 300 chargingRateUnit <Configured chargingRateUnit></p> |

| Tool validations |
|--|
| <p>* Step 2: (Message: SetChargingProfileResponse) status is <i>Accepted</i></p> <p>* Step 4: (Message: GetCompositeScheduleResponse) status <i>Accepted</i> evseld <i><Configured evseld></i> ChargingSchedule: duration <i>300</i> chargingRateUnit <i><Configured chargingRateUnit></i> scheduleStart <i><The time the GetCompositeScheduleRequest was transmitted +/- <Configured max time deviation>></i> <i>Note: The period of time between the <code>scheduleStart</code> from the SetChargingProfileRequest and the <code>scheduleStart</code> from the GetCompositeScheduleResponse is called x.</i> startPeriod <i>0</i>, limit <i><Local limit of Charging Station (> 6 * <limit multiplier>)></i> startPeriod <i>(60 - x)</i>, limit <i>6 * <limit multiplier></i> <i>Note: Check Determine Charging Profile Limit Multiplier for <limit multiplier></i></p> |
| <p>Post scenario validations: N/a</p> |

TC_K_24_CS: Clear Charging Profile - With stackLevel/purpose combination for multiple profiles

| | |
|--------------------------|--|
| Test case name | Clear Charging Profile - With stackLevel/purpose combination for multiple profiles |
| Test case Id | TC_K_24_CS |
| Use case Id(s) | K10 |
| Requirement(s) | K10.FR.04 |
| System under test | Charging Station |
| Description | If the CSMS wishes to clear some or all of the charging profiles that were previously sent to the Charging Station, then the CSMS sends a ClearChargingProfileRequest to the Charging Station. |
| Purpose | To verify if the Charging station is able to accept the request and clear charging profiles sent with a stackLevel/purpose combination by the CSMS as described at the OCPP specification. |
| Prerequisite(s) | Charging Station needs to have 2 or more EVSE. |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: A chargingprofile with <Configured chargingProfilePurpose> AND <Configured stackLevel> is configured for evseld 1. A chargingprofile with <Configured chargingProfilePurpose> AND <Configured stackLevel> is configured for evseld 2. |
| Reusable State: N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a ClearChargingProfileResponse | 1. The Test System sends a ClearChargingProfileRequest with chargingProfileCriteria.chargingProfilePurpose <Configured chargingProfilePurpose> AND chargingProfileCriteria.stackLevel <Configured stackLevel> |
| 4. The Charging Station responds with a GetChargingProfilesResponse | 3. The Test System sends a GetChargingProfilesRequest with chargingProfile.chargingProfilePurpose <Configured chargingProfilePurpose> AND chargingProfile.stackLevel <Configured stackLevel> |

| |
|---|
| Tool validations |
| * Step 2: Message ClearChargingProfileResponse - status <i>Accepted</i> * Step 4: Message GetChargingProfilesResponse - status <i>NoProfiles</i> |
| Post scenario validations: - N/a |

TC_K_28_CS: Set Charging Profile - TxDefaultProfile with transaction ongoing

| | |
|-------------------|---|
| Test case name | Set Charging Profile - TxDefaultProfile with transaction ongoing |
| Test case Id | TC_K_28_CS |
| Use case Id(s) | K01 |
| Requirement(s) | K01.FR.32 |
| System under test | Charging Station |
| Description | The CSMS sets a default schedule for a currently ongoing transaction. |
| Purpose | To verify if the CSMS and Charging Station are able to exchange messages to set a default schedule for a currently ongoing transaction. |
| Prerequisite(s) | N/a |

Before (Preparations)

Configuration State:
SmartChargingCtrlr.LimitChangeSignificance is 1.0

Memory State:
 N/a

Reusable State(s):
 State is *EnergyTransferStarted*

Main (Test scenario)

| Charging Station | CSMS |
|---|---|
| 2. The Charging Station responds with a SetChargingProfileResponse | <p>1. The Test System sends a SetChargingProfileRequest with chargingProfile.chargingProfilePurpose is <i>TxDefaultProfile</i> chargingProfile.chargingProfileKind is <i>Absolute</i> chargingProfile.chargingSchedule[0].duration is $<400 + \text{<Configured max time deviation> seconds}>$ evseld <i><Configured evseld></i> chargingProfile.chargingSchedule[0].startSchedule <i><current dateTime - <Configured max time deviation> seconds></i> chargingProfile.chargingSchedule[0].chargingSchedulePeriod[0].numberPhases <i><Configured numberPhases></i> chargingProfile.chargingSchedule[0].chargingSchedulePeriod[0].startPeriod 0 chargingProfile.chargingSchedule[0].chargingSchedulePeriod[0].limit $6 * \text{<limit multiplier>}$ <i>Note: Check Determine Charging Profile Limit Multiplier for <limit multiplier></i></p> |
| 4. The Charging Station responds with a GetCompositeScheduleResponse | <p>3. The Test System sends a GetCompositeScheduleRequest with evseld <i><Configured evseld></i> duration is 300 chargingRateUnit <i><Configured chargingRateUnit></i></p> |

| Tool validations |
|---|
| <p>* Step 2: (Message: SetChargingProfileResponse) status is <i>Accepted</i></p> <p>* Step 4: (Message: GetCompositeScheduleResponse) status <i>Accepted</i> evseld <i><Configured evseld></i> ChargingSchedule: duration <i>300</i> chargingRateUnit <i><Configured chargingRateUnit></i> scheduleStart <i><The time the GetCompositeScheduleRequest was transmitted +/- <Configured max time deviation>></i> startPeriod <i>0</i>, limit <i>6 * <limit multiplier></i> Note: Check Determine Charging Profile Limit Multiplier for <i><limit multiplier></i></p> |
| <p>Post scenario validations: N/a</p> |

TC_K_29_CS: Get Charging Profile - Evseld 0

| | |
|-------------------|---|
| Test case name | Get Charging Profile - Evseld 0 |
| Test case Id | TC_K_29_CS |
| Use case Id(s) | K09 |
| Requirement(s) | K09.FR.02, K09.FR.05 |
| System under test | Charging Station |
| Description | With the GetChargingProfilesRequest message the CSMS can ask a Charging Station to report all, or a subset of all the install Charging Profiles from the different possible sources. This can be used for some automatic smart charging control system, or for debug purposes by a CSO. |
| Purpose | To verify if the Charging station is able to successfully report the charging profile(s) requested as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: Charging station has a charging profile with <Generated Id1> AND chargingProfilePurpose <i>ChargingStationMaxProfile</i> configured on the charging station. Charging station has a second charge profile with <Generated Id2> AND chargingProfilePurpose <i>TxDefaultProfile</i> configured on <Configured evseld>. |
| Reusable State: N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a GetChargingProfilesResponse | 1. The Test System sends a GetChargingProfilesRequest with evseld 0 |
| 3. The Charging Station sends a ReportChargingProfilesRequest | 4. The Test System responds with a ReportChargingProfilesResponse |
| Note(s): - If tbc is True at Step 3 then step 3 and 4 will be repeated | |

| |
|--|
| Tool validations |
| * Step 2: Message GetChargingProfilesResponse - status <i>Accepted</i> * Step 3: Message ReportChargingProfilesRequest - requestId <Generated requestId> - chargingProfile <Generated ChargingProfile1> with chargingProfilePurpose <i>ChargingStationMaxProfile</i> |
| Post scenario validations: - All report message have been received |

TC_K_30_CS: Get Charging Profile - Evseld > 0

| | |
|-------------------|---|
| Test case name | Get Charging Profile - Evseld > 0 |
| Test case Id | TC_K_30_CS |
| Use case Id(s) | K09 |
| Requirement(s) | K09.FR.02, K09.FR.04 |
| System under test | Charging Station |
| Description | With the GetChargingProfilesRequest message the CSMS can ask a Charging Station to report all, or a subset of all the install Charging Profiles from the different possible sources. This can be used for some automatic smart charging control system, or for debug purposes by a CSO. |
| Purpose | To verify if the Charging station is able to successfully report the charging profile(s) requested for a specific EVSE as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: Charging station has a charging profile with <Generated Id1> AND <i>ChargingStationMaxProfile</i> configured on the charging station. Charging station has a second charge profile with <Generated Id2> AND <i>TxDefaultProfile</i> configured on <Configured evseld>. |
| Reusable State: N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a GetChargingProfilesResponse | 1. The Test System sends a GetChargingProfilesRequest with evseld <Configured evseld> |
| 3. The Charging Station sends a ReportChargingProfilesRequest | 4. The Test System responds with a ReportChargingProfilesResponse |
| Note(s): - If tbc is True at Step 3 then step 3 and 4 will be repeated | |

| |
|---|
| Tool validations |
| * Step 2: Message GetChargingProfilesResponse - status <i>Accepted</i> * Step 3: Message ReportChargingProfilesRequest - requestId <Generated requestId> - chargingProfile <Generated ChargingProfile> |
| Post scenario validations: - All report message have been received |

TC_K_31_CS: Get Charging Profile - No Evseld

| | |
|-------------------|---|
| Test case name | Get Charging Profile - No Evseld |
| Test case Id | TC_K_31_CS |
| Use case Id(s) | K09 |
| Requirement(s) | K09.FR.02, K09.FR.06 |
| System under test | Charging Station |
| Description | With the GetChargingProfilesRequest message the CSMS can ask a Charging Station to report all, or a subset of all the install Charging Profiles from the different possible sources. This can be used for some automatic smart charging control system, or for debug purposes by a CSO. |
| Purpose | To verify if the Charging station is able to successfully report all installed charging profiles requested as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: Charging station has a charging profile with <Generated Id1> AND ChargingStationMaxProfile configured on the charging station. Charging station has a second charge profile with <Generated Id2> AND TxDefaultProfile configured on EVSEId <Configured evseld>. |
| Reusable State: N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a GetChargingProfilesResponse | 1. The Test System sends a GetChargingProfilesRequest with: requestId <i>Generated requestId</i> |
| 3. The Charging Station sends a ReportChargingProfilesRequest | 4. The Test System responds with a ReportChargingProfilesResponse |
| Note(s): - If tbc is True at Step 3 then step 3 and 4 will be repeated | |

| |
|--|
| Tool validations |
| * Step 2: Message GetChargingProfilesResponse - status <i>Accepted</i> * Step 3: Message ReportChargingProfilesRequest - requestId <Generated requestId> - chargingProfiles <Configured ChargingProfiles> |
| Post scenario validations: - All report message have been received |

TC_K_32_CS: Get Charging Profile - chargingProfileId

| | |
|-------------------|---|
| Test case name | Get Charging Profile - chargingProfileId |
| Test case Id | TC_K_32_CS |
| Use case Id(s) | K09 |
| Requirement(s) | K09.FR.01, K09.FR.02 |
| System under test | Charging Station |
| Description | With the GetChargingProfilesRequest message the CSMS can ask a Charging Station to report all, or a subset of all the install Charging Profiles from the different possible sources. This can be used for some automatic smart charging control system, or for debug purposes by a CSO. |
| Purpose | To verify if the Charging station is able to successfully report a specific charging profile requested as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: Charging station has a charging profile with <Generated Id1> AND ChargingStationMaxProfile configured on the charging station. Charging station has a second charge profile with <Generated Id2> AND TxDefaultProfile configured on EVSEId <Configured evseId>. |
| Reusable State: N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a GetChargingProfilesResponse | 1. The Test System sends a GetChargingProfilesRequest with chargingProfileId <Generated Id1> |
| 3. The Charging Station sends a ReportChargingProfilesRequest | 4. The Test System responds with a ReportChargingProfilesResponse |
| Note(s): - If tbc is True at Step 3 then step 3 and 4 will be repeated | |

| |
|---|
| Tool validations |
| * Step 2: Message GetChargingProfilesResponse - status Accepted * Step 3: Message ReportChargingProfilesRequest - requestId Generated Id1 - chargingProfile <Configured chargingProfile> |
| Post scenario validations: - All report message have been received |

TC_K_33_CS: Get Charging Profile - Evseld > 0 + stackLevel

| | |
|-------------------|---|
| Test case name | Get Charging Profile - Evseld > 0 + stackLevel |
| Test case Id | TC_K_33_CS |
| Use case Id(s) | K09 |
| Requirement(s) | K09.FR.02, K09.FR.04 |
| System under test | Charging Station |
| Description | With the GetChargingProfilesRequest message the CSMS can ask a Charging Station to report all, or a subset of all the install Charging Profiles from the different possible sources. This can be used for some automatic smart charging control system, or for debug purposes by a CSO. |
| Purpose | To verify if the Charging station is able to successfully report a charging profile with specific stackLevel requested for a specific EVSE as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: Charging station has a charging profile with <Generated Id1> AND ChargingStationMaxProfile AND <Configured stackLevel> configured on the station. Charging station has a second charge profile with <Generated Id2> AND TxDefaultProfile AND <Configured stackLevel2> configured on <Configured evseld>. |
| Reusable State: N/a |

| Main (Test scenario) | |
|---|---|
| Charging Station | CSMS |
| 2. The Charging Station responds with a GetChargingProfilesResponse | 1. The Test System sends a GetChargingProfilesRequest with evseld <Configured evseld> AND chargingProfile.stackLevel <Configured stackLevel> |
| 3. The Charging Station sends a ReportChargingProfilesRequest | 4. The Test System responds with a ReportChargingProfilesResponse |
| <u>Note(s):</u> - If tbc is True at Step 3 then step 3 and 4 will be repeated | |

| |
|---|
| Tool validations |
| * Step 2: Message GetChargingProfilesResponse - status Accepted * Step 3: Message ReportChargingProfilesRequest - requestId Generated Id1 - chargingProfile <Configured ChargingProfile> |
| Post scenario validations: - All report message have been received |

TC_K_34_CS: Get Charging Profile - Evseld > 0 + chargingLimitSource

| | |
|-------------------|---|
| Test case name | Get Charging Profile - Evseld > 0 + chargingLimitSource |
| Test case Id | TC_K_34_CS |
| Use case Id(s) | K09 |
| Requirement(s) | K09.FR.02, K09.FR.04 |
| System under test | Charging Station |
| Description | With the GetChargingProfilesRequest message the CSMS can ask a Charging Station to report all, or a subset of all the install Charging Profiles from the different possible sources. This can be used for some automatic smart charging control system, or for debug purposes by a CSO. |
| Purpose | To verify if the Charging station is able to successfully report a charging profile with specific chargingLimitSource requested for a specific EVSE as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: <Configured chargingLimitSource> should be CSO AND <Configured chargingLimitSource2> should have no existing profiles AND Charging station has a charging profile with: - id <Generated Id1> - chargingProfilePurpose TxDefaultProfile - stackLevel <Configured StackLevel + 1> |
| Reusable State: N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a GetChargingProfilesResponse | 1. The Test System sends a GetChargingProfilesRequest with evseld <Configured evseld> AND chargingProfile.chargingLimitSource <Configured chargingLimitSource> |
| 3. The Charging Station sends a ReportChargingProfilesRequest | 4. The Test System responds with a ReportChargingProfilesResponse |
| Note(s): - If tbc is True at Step 3 then step 3 and 4 will be repeated | |
| 6. The Charging Station responds with a GetChargingProfilesResponse | 5. The Test System sends a GetChargingProfilesRequest with evseld <Configured evseld> AND chargingProfile.chargingLimitSource <Configured chargingLimitSource2> |

| |
|---|
| Tool validations |
| * Step 2: Message GetChargingProfilesResponse - status Accepted * Step 3: Message ReportChargingProfilesRequest - requestId Generated Id1 - chargingProfile <ChargingProfile> * Step 6: Message GetChargingProfilesResponse - status NoProfiles |

| |
|--|
| Tool validations |
| Post scenario validations: - All report message have been received |

TC_K_35_CS: Get Charging Profile - Evseld > 0 + chargingProfilePurpose

| | |
|-------------------|---|
| Test case name | Get Charging Profile - Evseld > 0 + chargingProfilePurpose |
| Test case Id | TC_K_35_CS |
| Use case Id(s) | K09 |
| Requirement(s) | K09.FR.02, K09.FR.04 |
| System under test | Charging Station |
| Description | With the GetChargingProfilesRequest message the CSMS can ask a Charging Station to report all, or a subset of all the install Charging Profiles from the different possible sources. This can be used for some automatic smart charging control system, or for debug purposes by a CSO. |
| Purpose | To verify if the Charging station is able to successfully report a charging profile with specific chargingProfilePurpose requested for a specific EVSE as described at the OCPP specification. |
| Prerequisite(s) | n/a |

Before (Preparations)

Configuration State:

N/a

Memory State:

Charging station has a charge profile with <Generated Id1> AND ChargingStationMaxProfile configured on the charging station.
Charging station has a second charge profile with <Generated Id2> AND TxDefaultProfile configured on <Configured evseld>.

Reusable State:

State is *EnergyTransferStarted*

Main (Test scenario)

| Charging Station | CSMS |
|--|---|
| 2. The Charging Station responds with a GetChargingProfilesResponse | 1. The Test System sends a GetChargingProfilesRequest with evseld <Configured evseld> AND chargingProfile.chargingProfilePurpose <Configured chargingProfilePurpose> |
| 3. The Charging Station sends a ReportChargingProfilesRequest | 4. The Test System responds with a ReportChargingProfilesResponse |

Note(s):

- If **tbc** is True at Step 3 then step 3 and 4 will be repeated

Tool validations

* Step 2:

Message **GetChargingProfilesResponse**- **status** *Accepted*

* Step 3:

Message **ReportChargingProfilesRequest**- **requestId** *Generated Id1*- **ChargingProfile** <Configured ChargingProfile>

Post scenario validations:

- All report message have been received

TC_K_36_CS: Get Charging Profile - Evseld > 0 + chargingProfilePurpose + stackLevel

| | |
|-------------------|---|
| Test case name | Get Charging Profile - Evseld > 0 + chargingProfilePurpose + stackLevel |
| Test case Id | TC_K_36_CS |
| Use case Id(s) | K09 |
| Requirement(s) | K09.FR.02, K09.FR.04 |
| System under test | Charging Station |
| Description | With the GetChargingProfilesRequest message the CSMS can ask a Charging Station to report all, or a subset of all the install Charging Profiles from the different possible sources. This can be used for some automatic smart charging control system, or for debug purposes by a CSO. |
| Purpose | To verify if the Charging station is able to successfully report a charging profile with specific chargingProfilePurpose and stackLevel requested for a specific EVSE as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: Charging station has a charge profile with <Generated Id1> AND ChargingStationMaxProfile AND <Configured stackLevel> configured on the charging station. Charging station has a second charge profile with <Generated Id2> AND TxDefaultProfile AND <Configured stackLevel> configured on <Configured evseld>. |
| Reusable State: N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a GetChargingProfilesResponse | 1. The Test System sends a GetChargingProfilesRequest with evseld <Configured evseld> AND chargingProfile.chargingProfilePurpose <TxDefaultProfile> AND chargingProfile.stackLevel <Configured stackLevel> |
| 3. The Charging Station sends a ReportChargingProfilesRequest | 4. The Test System responds with a ReportChargingProfilesResponse |
| Note(s): - If tbc is True at Step 3 then step 3 and 4 will be repeated | |

| |
|---|
| Tool validations |
| * Step 2: Message GetChargingProfilesResponse - status Accepted * Step 3: Message ReportChargingProfilesRequest - requestId Generated Id1 - ChargingProfile <Configured ChargingProfile> |
| Post scenario validations: - All report message have been received |

TC_K_60_CS: Set Charging Profile - TxProfile with ongoing transaction on the specified EVSE

| | |
|-------------------|---|
| Test case name | Set Charging Profile - TxProfile with ongoing transaction on the specified EVSE |
| Test case Id | TC_K_60_CS |
| Use case Id(s) | K01 |
| Requirement(s) | K01.FR.04, K01.FR.07, K01.FR.15 |
| System under test | Charging Station |
| Description | The CSMS sets a TxProfile on a specific EVSE for a currently ongoing transaction. |
| Purpose | To verify if the Charging Station is able to exchange messages to set a TxProfile on a specific EVSE for a currently ongoing transaction. |
| Prerequisite(s) | The Charging Station must support the GetChargingProfiles feature. |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a SetChargingProfileResponse | <p>1. The Test System sends a SetChargingProfileRequest with chargingProfile.chargingProfilePurpose is <i>TxProfile</i> chargingProfile.transactionId is <i><transactionId returned by Charging Station in before></i> chargingProfile.chargingProfileKind is <i>Relative</i> evseld <i><Configured evseld></i> chargingProfile.chargingSchedule[0].chargingSchedulePeriod[0].numberPhases <i><Configured numberPhases></i> chargingProfile.chargingSchedule[0].chargingSchedulePeriod[0].startPeriod <i>0</i> chargingProfile.chargingSchedule[0].chargingSchedulePeriod[0].limit <i>6 * <limit multiplier></i> Note: Check <i>Determine Charging Profile Limit Multiplier</i> for <i><limit multiplier></i></p> |
| 4. The Charging Station responds with a GetChargingProfilesResponse | <p>3. The Test System sends a GetChargingProfilesRequest with chargingProfile.chargingProfileId <i><Used chargingProfileId at step 1></i></p> |
| 5. The Charging Station sends a ReportChargingProfilesRequest | <p>6. The Test System responds with a ReportChargingProfilesResponse</p> |

| |
|--|
| Tool validations |
| <p>* Step 2: (Message: SetChargingProfileResponse) status is <i>Accepted</i></p> <p>* Step 4: (Message: GetChargingProfilesResponse) status is <i>Accepted</i></p> <p>* Step 5: (Message: ReportChargingProfilesRequest) chargingProfile <i><The Charging Profile set at step 1></i></p> |

| |
|--|
| Tool validations |
| Post scenario validations: N/a |

TC_K_37_CS: Remote start transaction with charging profile - Success

| | |
|-------------------|---|
| Test case name | Remote start transaction with charging profile - Success |
| Test case Id | TC_K_37_CS |
| Use case Id(s) | K05,F01 |
| Requirement(s) | K05.FR.03, E01.FR.02,F01.FR.10,F01.FR.13 |
| System under test | Charging Station |
| Description | The CSMS sets a TxProfile on a specific EVSE inside a RequestStartTransactionRequest message. |
| Purpose | To verify if the Charging Station is able to set a TxProfile on a specific EVSE when receiving one in a RequestStartTransactionRequest message. |
| Prerequisite(s) | The Charging Station must support the GetChargingProfiles feature. |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State(s):

State is *EVConnectedPreSession*

Main (Test scenario)

| Charging Station | CSMS |
|---|--|
| 2. The Charging Station responds with a RequestStartTransactionResponse | 1. The Test System sends a RequestStartTransactionRequest with idToken.idToken <Configured valid_idtoken_idtoken> idToken.type <Configured valid_idtoken_type> evseld <Configured evseld> chargingProfile.chargingProfilePurpose is <i>TxProfile</i> chargingProfile.transactionId is omitted. chargingProfile.chargingProfileKind is <i>Relative</i> chargingProfile.chargingSchedule[0].chargingSchedulePeriod[0].numberPhases <Configured numberPhases> chargingProfile.chargingSchedule[0].chargingSchedulePeriod[0].startPeriod 0 chargingProfile.chargingSchedule[0].chargingSchedulePeriod[0].limit 6 * <limit multiplier> Note: Check <i>Determine Charging Profile Limit Multiplier</i> for <limit multiplier> |
| 3. The Charging Station sends an AuthorizeRequest <u>Note(s):</u> - This step needs to be executed when AuthCtrlr.AuthorizeRemoteStart is true, unless (AuthEnabled is implemented with mutability <i>ReadOnly</i> AND the value is set to false) OR the idToken is cached. In case the idToken is used for a reservation, sending the AuthorizeRequest message is optional. | 4. The Test System responds with an AuthorizeResponse with idTokenInfo.status <i>Accepted</i> |
| 5. The Charging Station sends a TransactionEventRequest <u>Note(s):</u> - This step needs to be executed when TxStartPoint contains <i>Authorized</i> OR the transaction already started. So in the case TxStartPoint contains <i>ParkingBayOccupancy</i> or (<i>EVConnected</i> , in the case this testcase was initiated from state <i>EVConnectedPreSession</i> .) | 6. The Test System responds with a TransactionEventResponse <u>Note(s):</u> - The first TransactionEventRequest sent after authorization contains the idToken field. The TransactionEventResponse of this request message contains idTokenInfo with status <i>Accepted</i> |

| Main (Test scenario) | |
|--|---|
| 7. Execute Reusable State <i>EnergyTransferStarted</i> | |
| 9. The Charging Station responds with a GetChargingProfilesResponse | 8. The Test System sends a GetChargingProfilesRequest with chargingProfile.chargingProfileId <Used chargingProfileId at step 1> |
| 10. The Charging Station sends a ReportChargingProfilesRequest | 11. The Test System responds with a ReportChargingProfilesResponse |

| Tool validations |
|---|
| <p>* Step 2:</p> <p>Message: RequestStartTransactionResponse</p> <ul style="list-style-type: none"> - status must be <i>Accepted</i> <p>If the transaction has already been started, so if TxStartPoint contains <i>ParkingBayOccupancy</i> OR (<Configured TxStartPoint> contains <i>EVConnected</i> AND State pre reusable state execution was <i>EVConnectedPreSession</i>) then</p> <ul style="list-style-type: none"> - transactionId must be <Provided transactionId in first TransactionEventRequest> <p>* Step 3:</p> <p>Message: AuthorizeRequest</p> <ul style="list-style-type: none"> - idToken.idToken <Configured valid_idtoken_idtoken> - idToken.type <Configured valid_idtoken_type> <p>* Step 5:</p> <p>Message: TransactionEventRequest</p> <ul style="list-style-type: none"> - triggerReason must be <i>RemoteStart</i> - transactionInfo.remoteStartId must be present. <p>* Step 9:</p> <p>(Message: GetChargingProfilesResponse)</p> <p>status is <i>Accepted</i></p> <p>* Step 10:</p> <p>(Message: ReportChargingProfilesRequest)</p> <p>chargingProfile <The Charging Profile set at step 1></p> |
| <p>Post scenario validations:</p> <p>N/a</p> |

TC_K_38_CS: Remote start transaction with charging profile - Ignore chargingProfile

| | |
|-------------------|---|
| Test case name | Remote start transaction with charging profile - Ignore chargingProfile |
| Test case Id | TC_K_38_CS |
| Use case Id(s) | F01 |
| Requirement(s) | F01.FR.12,F01.FR.13 |
| System under test | Charging Station |
| Description | The CSMS sets a TxProfile on a specific EVSE inside a RequestStartTransactionRequest message. |
| Purpose | To verify if the Charging Station is able to ignore a TxProfile on a specific EVSE when receiving one in a RequestStartTransactionRequest message, when it does not support Smart Charging. |
| Prerequisite(s) | The Charging Station does NOT support Smart Charging. |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EVConnectedPreSession</i> |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a RequestStartTransactionResponse | <p>1. The Test System sends a RequestStartTransactionRequest with idToken.idToken <Configured valid_idtoken_idtoken> idToken.type <Configured valid_idtoken_type> evseld <Configured evseld> chargingProfile.chargingProfilePurpose is TxProfile chargingProfile.transactionId is omitted. chargingProfile.chargingProfileKind is Relative chargingProfile.chargingSchedule.chargingSchedulePeriod[0].numberPhases <Configured numberPhases> chargingProfile.chargingSchedule.chargingSchedulePeriod[0].startPeriod 0 chargingProfile.chargingSchedule[0].chargingSchedulePeriod[0].limit 6 * <limit multiplier> Note: Check <i>Determine Charging Profile Limit Multiplier</i> for <limit multiplier></p> |
| <p>3. The Charging Station sends an AuthorizeRequest</p> <p><u>Note(s):</u> - This step needs to be executed when AuthCtrlr.AuthorizeRemoteStart is true, unless (AuthEnabled is implemented with mutability ReadOnly AND the value is set to false) OR the idToken is cached. In case the idToken is used for a reservation, sending the AuthorizeRequest message is optional.</p> | <p>4. The Test System responds with an AuthorizeResponse with idTokenInfo.status Accepted</p> |

| Main (Test scenario) | |
|---|---|
| <p>5. The Charging Station sends a TransactionEventRequest</p> <p><u>Note(s):</u> - This step needs to be executed when TxStartPoint contains Authorized OR the transaction already started. So in the case TxStartPoint contains <i>ParkingBayOccupancy</i> or (<i>EVConnected</i>, in the case this testcase was initiated from state <i>EVConnectedPreSession</i>.)</p> | <p>6. The Test System responds with a TransactionEventResponse</p> <p><u>Note(s):</u> - The first <i>TransactionEventRequest</i> sent after authorization contains the <i>idToken</i> field. The <i>TransactionEventResponse</i> of this request message contains idTokenInfo with status Accepted</p> |
| <p>7. Execute Reusable State <i>EnergyTransferStarted</i></p> | |

| Tool validations |
|---|
| <p>* Step 2: Message: RequestStartTransactionResponse - status must be <i>Accepted</i> If the transaction has already been started, so if <i>TxStartPoint</i> contains <i>ParkingBayOccupancy</i> OR (<i>TxStartPoint</i> contains <i>EVConnected</i> AND State pre reusable state execution was <i>EVConnectedPreSession</i>) then - transactionId must be <i><Provided transactionId in first TransactionEventRequest></i></p> <p>* Step 3: Message: AuthorizeRequest - idToken.idToken <i><Configured valid_idtoken_idtoken></i> - idToken.type <i><Configured valid_idtoken_type></i></p> <p>* Step 5: Message: TransactionEventRequest - triggerReason must be <i>RemoteStart</i> - transactionInfo.remoteStartId must be present.</p> |
| <p>Post scenario validations: N/a</p> |

TC_K_39_CS: Get Composite Schedule - No ChargingProfile installed on Charging Station

| | |
|-------------------|---|
| Test case name | Get Composite Schedule - No ChargingProfile installed on Charging Station |
| Test case Id | TC_K_39_CS |
| Use case Id(s) | K08 |
| Requirement(s) | K08.FR.02, K08.FR.03,K08.FR.06 |
| System under test | Charging Station |
| Description | The CSMS requests a composite schedule which is a combination of local limits and the prevailing Charging Profiles of the different chargingProfilePurposes and stack levels. |
| Purpose | To verify if the Charging Station is able to calculate a correct composite schedule and provide this to the CSMS on request. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|---|---|
| Charging Station | CSMS |
| 2. The Charging Station responds with a GetCompositeScheduleResponse | 1. The Test System sends a GetCompositeScheduleRequest with evseld 0 duration is 300 chargingRateUnit <Configured chargingRateUnit> |

| |
|--|
| Tool validations |
| <p>* Step 2: (Message: GetCompositeScheduleResponse) status Accepted evseld 0 scheduleStart <The time the GetCompositeScheduleRequest was transmitted +/- <Configured max time deviation>> duration is 300 chargingRateUnit <Configured chargingRateUnit> startPeriod 0</p> |
| Post scenario validations: N/a |

TC_K_40_CS: Get Composite Schedule - Stacking ChargingProfiles

| | |
|-------------------|---|
| Test case name | Get Composite Schedule - Stacking ChargingProfiles |
| Test case Id | TC_K_40_CS |
| Use case Id(s) | K08 |
| Requirement(s) | K08.FR.02,K08.FR.06 |
| System under test | Charging Station |
| Description | The CSMS requests a composite schedule which is a combination of local limits and the prevailing Charging Profiles of the different chargingProfilePurposes and stack levels. 2 ChargingProfiles with same startSchedule and different stackLevels are submitted. |
| Purpose | To verify if the Charging Station is able to calculate a correct composite schedule and provide this to the CSMS on request. |
| Prerequisite(s) | <ul style="list-style-type: none"> - ChargingProfileEntries.maxLimit must be > 1 - The configuration variable ChargingProfileMaxStackLevel must be > 0 - The configuration variable PeriodsPerSchedule must be > 2 |

Before (Preparations)

Configuration State:

N/a

Memory State:

set <startScheduleTime> = <current dateTime> - <Configured max time deviation>

[SetChargingProfile](#) with

ChargingProfile 1:

chargingProfilePurpose is *TxDefaultProfile*

chargingProfileKind should be *Absolute*

stackLevel should be 0

evseld <Configured evseld>

startSchedule <startScheduleTime>

numberPhases <Configured numberPhases>

ChargingSchedule:

duration 400

chargingRateUnit <Configured chargingRateUnit>

startPeriod 0, **limit** 6 * <limit multiplier>

startPeriod 100, **limit** 8 * <limit multiplier>

startPeriod 200, **limit** 10 * <limit multiplier>

Note: Check [Determine Charging Profile Limit Multiplier](#) for <limit multiplier>

ChargingProfile 2:

chargingProfilePurpose is *TxDefaultProfile*

chargingProfileKind should be *Absolute*

stackLevel should be 1

evseld <Configured evseld>

startSchedule <startScheduleTime>

numberPhases <Configured numberPhases>

ChargingSchedule:

duration 150

chargingRateUnit <Configured chargingRateUnit>

startPeriod 0, **limit** 7 * <limit multiplier>

Note: Check [Determine Charging Profile Limit Multiplier](#) for <limit multiplier>

startPeriod 100, **limit** 9 * <limit multiplier>

Note: Check [Determine Charging Profile Limit Multiplier](#) for <limit multiplier>

Reusable State(s):

N/a

| Main (Test scenario) | |
|---|---|
| Charging Station | CSMS |
| 2. The Charging Station responds with a GetCompositeScheduleResponse | 1. The Test System sends a GetCompositeScheduleRequest with evseld <Configured evseld> duration is 350 chargingRateUnit <Configured chargingRateUnit> |

| Tool validations |
|--|
| <p>* Step 2: (Message: GetCompositeScheduleResponse) status <i>Accepted</i> evseld <Configured evseld> ChargingSchedule: duration 350 chargingRateUnit <Configured chargingRateUnit> scheduleStart <The time the GetCompositeScheduleRequest was transmitted> plus/minus <Configured max time deviation> Note: The period of time between the <i>scheduleStart</i> from the SetChargingProfileRequest and the <i>scheduleStart</i> from the GetCompositeScheduleResponse is called x. startPeriod 0, limit $7 * \text{<limit multiplier>}$ (stackLevel 1) startPeriod (100 - x), limit $9 * \text{<limit multiplier>}$ (stackLevel 1) startPeriod (150 - x), limit $8 * \text{<limit multiplier>}$ (stackLevel 0) startPeriod (200 - x), limit $10 * \text{<limit multiplier>}$ (stackLevel 0) Note: Check Determine Charging Profile Limit Multiplier for <limit multiplier></p> |
| Post scenario validations: N/a |

TC_K_41_CS: Get Composite Schedule - Combining chargingProfilePurposes

| | |
|-------------------|--|
| Test case name | Get Composite Schedule - Combining chargingProfilePurposes |
| Test case Id | TC_K_41_CS |
| Use case Id(s) | K08 |
| Requirement(s) | K08.FR.02,K08.FR.04 |
| System under test | Charging Station |
| Description | The CSMS requests a composite schedule which is a combination of local limits and the prevailing Charging Profiles of the different chargingProfilePurposes and stack levels. 3 ChargingProfiles are submitted with the same startSchedule. |
| Purpose | To verify if the Charging Station is able to calculate a correct composite schedule and provide this to the CSMS on request. |
| Prerequisite(s) | - ChargingProfileEntries.maxLimit must be > 2 - The configuration variable PeriodsPerSchedule must be > 2 |

| | |
|--|---|
| Before (Preparations) | |
| Configuration State: N/a | |
| Memory State: set <startScheduleTime> = <current dateTime> - <Configured max time deviation> seconds Note: Set MaxProfile for the next 24 hours: SetChargingProfile with ChargingProfile 1: chargingProfilePurpose is <i>ChargingStationMaxProfile</i> chargingProfileKind should be <i>Absolute</i> stackLevel should be 0 evseld 0 startSchedule <startScheduleTime> numberPhases <Configured numberPhases> ChargingSchedule: duration 86400 chargingRateUnit <Configured chargingRateUnit> startPeriod 0, limit 10 * <limit multiplier> Note: Check Determine Charging Profile Limit Multiplier for <limit multiplier> | |
| Note: Set a default profile for 300 seconds ChargingProfile 2: chargingProfilePurpose is <i>TxDefaultProfile</i> chargingProfileKind should be <i>Absolute</i> stackLevel should be 0 evseld <Configured evseld> startSchedule <startScheduleTime> numberPhases <Configured numberPhases> ChargingSchedule: duration 300 chargingRateUnit <Configured chargingRateUnit> startPeriod 0,60,120,180,260, limit 6,10,8,15,8 * <limit multiplier> Note: Check Determine Charging Profile Limit Multiplier for <limit multiplier> | Note: set a TxProfile for 260 seconds: ChargingProfile 3: chargingProfilePurpose is <i>TxProfile</i> chargingProfileKind should be <i>Absolute</i> stackLevel should be 0 evseld <Configured evseld> startSchedule <startScheduleTime> numberPhases <Configured numberPhases> ChargingSchedule: duration 260 chargingRateUnit <Configured chargingRateUnit> startPeriod 0,50,140,200,240, limit 8,11,16,6,12 * <limit multiplier> Note: Check Determine Charging Profile Limit Multiplier for <limit multiplier> |
| Reusable State(s): State is EnergyTransferStarted | |

| | |
|-----------------------------|------|
| Main (Test scenario) | |
| Charging Station | CSMS |

| Main (Test scenario) | |
|---|--|
| 2. The Charging Station responds with a GetCompositeScheduleResponse | <p>1. The Test System sends a GetCompositeScheduleRequest with evseld <Configured evseld> duration is 400 chargingRateUnit <Configured chargingRateUnit></p> |

| Tool validations |
|--|
| <p>* Step 2: (Message: GetCompositeScheduleResponse) status Accepted evseld <Configured evseld> ChargingSchedule: duration 400 chargingRateUnit <Configured chargingRateUnit> scheduleStart <The time the GetCompositeScheduleRequest was transmitted +/- <Configured max time deviation>> Note: The period of time between the <i>scheduleStart</i> from the SetChargingProfileRequest and the <i>scheduleStart</i> from the GetCompositeScheduleResponse is called x. startPeriod 0, limit $8 * \text{<limit multiplier>}$ (TxProfile) startPeriod (50 - x), limit $10 * \text{<limit multiplier>}$ (ChargingStationMaxProfile) startPeriod (200 - x), limit $6 * \text{<limit multiplier>}$ (TxProfile) startPeriod (240 - x), limit $10 * \text{<limit multiplier>}$ (ChargingStationMaxProfile) startPeriod (260 - x), limit $8 * \text{<limit multiplier>}$ (TxDefaultProfile) startPeriod (300 - x), limit $10 * \text{<limit multiplier>}$ (ChargingStationMaxProfile) Note: Check Determine Charging Profile Limit Multiplier for <limit multiplier></p> |
| <p>Post scenario validations: N/a</p> |

TC_K_42_CS: Get Composite Schedule - chargingRateUnit not supported

| | |
|-------------------|--|
| Test case name | Get Composite Schedule - chargingRateUnit not supported |
| Test case Id | TC_K_42_CS |
| Use case Id(s) | K08 |
| Requirement(s) | K08.FR.07 |
| System under test | Charging Station |
| Description | The CSMS requests a composite schedule which is a combination of local limits and the prevailing Charging Profiles of the different chargingProfilePurposes and stack levels. |
| Purpose | To verify if the Charging Station is able to reject a GetCompositeScheduleRequest when it asks for a not supported chargingRateUnit. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The Charging Station does NOT support one of the chargingRateUnits; A or W. - The Test System chargingRateUnit configuration field contains the NOT supported chargingRateUnit. |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|---|---|
| Charging Station | CSMS |
| 2. The Charging Station responds with a GetCompositeScheduleResponse | 1. The Test System sends a GetCompositeScheduleRequest with evseld 0 duration is 300 chargingRateUnit <Configured unsupported chargingRateUnit> |

| |
|---|
| Tool validations |
| * Step 2: (Message: GetCompositeScheduleResponse) status <i>Rejected</i> schedule is omitted |
| Post scenario validations: N/a |

TC_K_47_CS: Get Composite Schedule - Unknown EVSEId

| | |
|-------------------|---|
| Test case name | Get Composite Schedule - Unknown EVSEId |
| Test case Id | TC_K_47_CS |
| Use case Id(s) | K08 |
| Requirement(s) | K08.FR.05 |
| System under test | Charging Station |
| Description | The CSMS requests a composite schedule which is a combination of local limits and the prevailing Charging Profiles of the different chargingProfilePurposes and stack levels. |
| Purpose | To verify if the Charging Station is able to reject a GetCompositeScheduleRequest when it asks for composite schedule for a unknown evseld. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|---|---|
| Charging Station | CSMS |
| 2. The Charging Station responds with a GetCompositeScheduleResponse | 1. The Test System sends a GetCompositeScheduleRequest with evseld <Configured number of evse> + 1 duration is 300 chargingRateUnit <Configured chargingRateUnit> |

| |
|---|
| Tool validations |
| * Step 2: (Message: GetCompositeScheduleResponse) status <i>Rejected</i> schedule is omitted |
| Post scenario validations: N/a |

TC_K_52_CS: EMS Control - Set / Update External Charging Limit (not on a transaction) - ChargingStationExternalConstraints in report

| | |
|-------------------|--|
| Test case name | EMS Control - Set / Update External Charging Limit (not on a transaction) - ChargingStationExternalConstraints in report |
| Test case Id | TC_K_52_CS |
| Use case Id(s) | K12 |
| Requirement(s) | K12.FR.05 |
| System under test | Charging Station |
| Description | A charging schedule or charging limit has been set by an external system on the Charging Station. Such a charging limit is represented by a charging profile with purpose <i>ChargingStatioExternalConstraints</i> . |
| Purpose | To verify if the charging station is able to correctly report an external charging limit as <i>ChargingStationExternalConstraints</i> . |
| Prerequisite(s) | n/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: An external charging limit has been submitted to Charging Station. |
| Reusable State: N/a |

| Main (Test scenario) | |
|---|--|
| Charging Station | CSMS |
| 2. The Charging Station responds with a GetChargingProfilesResponse | 1. The Test System sends a GetChargingProfilesRequest with chargingProfile.chargingProfilePurpose <i>ChargingStationExternalConstraints</i> |
| 3. The Charging Station sends a ReportChargingProfilesRequest | 4. The Test System responds with a ReportChargingProfilesResponse |
| <u>Note(s):</u> - If tbc is True at Step 3 then step 3 and 4 will be repeated | |

| |
|---|
| Tool validations |
| * Step 2: Message GetChargingProfilesResponse - status <i>Accepted</i> |
| * Step 3: Message ReportChargingProfilesRequest - requestId <i>Same id as in the request in step 1</i> - chargingProfile.chargingProfilePurpose <i>ChargingStationExternalConstraints</i> |
| Post scenario validations: - All report messages have been received and at least one <i>ChargingStationExternalConstraints</i> is returned. |

TC_K_53_CS: Charging with load leveling based on High Level Communication - Success

| | |
|-------------------|--|
| Test case name | Charging with load leveling based on High Level Communication - Success |
| Test case Id | TC_K_53_CS |
| Use case Id(s) | K15 |
| Requirement(s) | K15.FR.01,K15.FR.06,K15.FR.09,K15.FR.10 |
| System under test | Charging Station |
| Description | ISO15118-1 E1 AC Charging with load leveling based on High Level Communication, and E4 DC charging with load leveling based on High Level Communication. |
| Purpose | To verify if the Charging Station is able to perform load leveling when it receives charging needs from the EV. |
| Prerequisite(s) | N/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EVConnectedPreSession</i> |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>Authorized15118</i> | |
| 2. The Charging Station sends a NotifyEVChargingNeedsRequest | 3. The Test System responds with a NotifyEVChargingNeedsResponse with status <i>Accepted</i> |
| 5. The Charging Station responds with a SetChargingProfileResponse | 4. The Test System sends a SetChargingProfileRequest with chargingProfile.chargingProfilePurpose <i>TxProfile</i> chargingProfile.transactionId <i><transactionId></i> chargingProfile.chargingSchedule[0].id <i><Id generated by Test System></i> chargingProfile.chargingSchedule[0].chargingRateUnit <i><Configured chargingRateUnit></i> chargingProfile.chargingSchedule[0].chargingSchedulePeriod.startPeriod <i>0</i> chargingProfile.chargingSchedule[0].chargingSchedulePeriod.limit <i>6 * <limit multiplier></i> <i>Note: Check Determine Charging Profile Limit Multiplier for <limit multiplier></i> |
| 6. The Charging Station sends a NotifyEVChargingScheduleRequest | 7. The Test System responds with a NotifyEVChargingScheduleResponse with status <i>Accepted</i> |
| 8. The Charging Station sends a TransactionEventRequest . | 9. The Test System responds with a TransactionEventResponse . |

| Tool validations |
|---|
| <p>* Step 1:</p> <p>Message: TransactionEventRequest of Authorized15118 step 3</p> <ul style="list-style-type: none"> - eventType must be <i>Updated</i> or <i>Started</i> - triggerReason must be <i>Authorized</i> <p>* Step 2:</p> <p>Message: NotifyEVChargingNeedsRequest</p> <p>IF chargingNeeds.acChargingParameters is <omitted> THEN chargingNeeds.dcChargingParameters must be <not omitted> END IF</p> <p>IF chargingNeeds.dcChargingParameters is <omitted> THEN chargingNeeds.acChargingParameters must be <not omitted> END IF</p> <p>* Step 5:</p> <p>Message: SetChargingProfileResponse</p> <ul style="list-style-type: none"> - status must be <i>Accepted</i> <p>* Step 6:</p> <p>Message: NotifyEVChargingScheduleRequest</p> <ul style="list-style-type: none"> - chargingSchedule must be within bounds of chargingSchedule of step 4 <p>* Step 8:</p> <p>Message: TransactionEventRequest</p> <ul style="list-style-type: none"> - triggerReason must be <i>ChargingStateChanged</i> - transactionInfo.chargingState must be <i>Charging</i> |
| <p>Post scenario validations:</p> <p>N/a</p> |

TC_K_54_CS: Charging with load leveling based on High Level Communication - No SASchedule (rejected)

| | |
|-------------------|--|
| Test case name | Charging with load leveling based on High Level Communication - No SASchedule (rejected) |
| Test case Id | TC_K_54_CS |
| Use case Id(s) | K15, K17 |
| Requirement(s) | K15.FR.01,K17.FR.04 |
| System under test | Charging Station |
| Description | ISO15118-1 E1 AC Charging with load leveling based on High Level Communication, and E4 DC charging with load leveling based on High Level Communication. |
| Purpose | To verify if the Charging Station is able to handle a Rejected status from the CSMS in response to providing the EV charging needs. |
| Prerequisite(s) | N/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EVConnectedPreSession</i> |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>Authorized15118</i> | |
| 2. The Charging Station sends a NotifyEVChargingNeedsRequest . | 3. The Test System responds with a NotifyEVChargingNeedsResponse . With status <i>Rejected</i> |
| 4. The Charging Station sends a NotifyEVChargingScheduleRequest . <u>Note(s)</u> : - <i>This step is optional. The Charging Station will only send it when the EV returns a charging profile.</i> | 5. The Test System responds with a NotifyEVChargingScheduleResponse . With status <i>Accepted</i> |
| 6. The Charging Station sends a TransactionEventRequest . | 7. The Test System responds with a TransactionEventResponse . |

| |
|---|
| Tool validations |
| <p>* Step 2: (Message: NotifyEVChargingNeedsRequest) evseld <Configured evseld></p> <p>* Step 4: (Message: NotifyEVChargingScheduleRequest) evseld <Configured evseld></p> <p>* Step 6: Message: TransactionEventRequest - triggerReason must be <i>ChargingStateChanged</i> - transactionInfo.chargingState must be <i>Charging</i></p> |
| Post scenario validations: N/a |

TC_K_56_CS: Charging with load leveling based on High Level Communication - Offline

| | |
|-------------------|--|
| Test case name | Charging with load leveling based on High Level Communication - Offline |
| Test case Id | TC_K_56_CS |
| Use case Id(s) | K15,K17 |
| Requirement(s) | K15.FR.15,K17.FR.15 |
| System under test | Charging Station |
| Description | ISO15118-1 E1 AC Charging with load leveling based on High Level Communication, and E4 DC charging with load leveling based on High Level Communication. |
| Purpose | To verify if the Charging Station is able to perform load leveling when it receives charging needs from the EV and it is offline. |
| Prerequisite(s) | N/a |

| |
|---|
| Before (Preparations) |
| <p>Configuration State: RetryBackOffWaitMinimum is <i><Configured RetryBackOffWaitMinimum></i></p> <p>Memory State: SetChargingProfile with ChargingProfile: chargingProfilePurpose is <i>TxDefaultProfile</i> chargingProfileKind should be <i>Absolute</i> stackLevel should be <i>0</i> evseld <i><Configured evseld></i> validFrom <i><current dateTime - <Configured max time deviation> seconds></i> validTo <i><current dateTime + <Configured max time deviation> + 401 seconds></i> startSchedule <i><current dateTime - <Configured max time deviation> seconds></i> numberPhases <i><Configured numberPhases></i></p> <p>ChargingSchedule: duration <i>400</i> chargingRateUnit <i><Configured chargingRateUnit></i> startPeriod <i>0</i>, limit <i>6 * <limit multiplier></i> Note: Check Determine Charging Profile Limit Multiplier for <i><limit multiplier></i></p> <p>Reusable State(s): State is EVConnectedPreSession State is Authorized15118</p> |

| Main (Test scenario) | |
|--|---|
| Charging Station | CSMS |
| | 1. The Test System closes the WebSocket connection AND does not accept a reconnect. |
| | 2. The Test System accepts the reconnection attempt from the Charging Station, after 90 seconds. |
| 3. The Charging Station sends a NotifyEVChargingScheduleRequest . <u>Note(s)</u> : - This step is optional. - It is allowed to execute this step either before or after the TransactionEventRequest from step 5. | 4. The Test System responds with a NotifyEVChargingScheduleResponse . With status Accepted |
| 5. The Charging Station sends a TransactionEventRequest . | 6. The Test System responds with a TransactionEventResponse . |

| Tool validations |
|--|
| <p>* Step 3: (Message: NotifyEVChargingScheduleRequest) evseld <Configured evseld></p> <p>* Step 5: Message: TransactionEventRequest</p> <ul style="list-style-type: none">- triggerReason must be <i>ChargingStateChanged</i>- transactionInfo.chargingState must be <i>Charging</i>- offline <i>true</i> |
| <p>Post scenario validations: N/a</p> |

TC_K_57_CS: Renegotiating a Charging Schedule - Initiated by EV

| | |
|-------------------|---|
| Test case name | Renegotiating a Charging Schedule - Initiated by EV |
| Test case Id | TC_K_57_CS |
| Use case Id(s) | K17 |
| Requirement(s) | K17.FR.01,K17.FR.06,K17.FR.09,K17.FR.10 |
| System under test | Charging Station |
| Description | The EV signals the Charging Station that it wants to renegotiate and it provides new charging needs, which the Charging Station sends to the CSMS. Based on this and other parameters, the CSMS calculates a new charging schedule and sends it via SetChargingProfileRequest to Charging Station, which communicates it to the EV. |
| Purpose | To verify if the Charging Station is able to perform load leveling when it receives a renegotiate request from the EV. |
| Prerequisite(s) | N/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EVConnectedPreSession</i> State is <i>Authorized15118</i> State is <i>EnergyTransferStarted</i> |

| | |
|--|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>RenegotiateChargingLimits</i> | |

| |
|-----------------------------------|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_K_58_CS: Renegotiating a Charging Schedule - Initiated by CSMS

| | |
|-------------------|---|
| Test case name | Renegotiating a Charging Schedule - Initiated by CSMS |
| Test case Id | TC_K_58_CS |
| Use case Id(s) | K17 |
| Requirement(s) | K17.FR.01,K17.FR.06,K17.FR.09,K17.FR.10 |
| System under test | Charging Station |
| Description | The EV signals the Charging Station that it wants to renegotiate and it provides new charging needs, which the Charging Station sends to the CSMS. Based on this and other parameters, the CSMS calculates a new charging schedule and sends it via SetChargingProfileRequest to Charging Station, which communicates it to the EV. |
| Purpose | To verify if the Charging Station is able to perform load leveling when it receives a renegotiate request from the CSMS. |
| Prerequisite(s) | N/a |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State(s):

State is *EVConnectedPreSession*State is *Authorized15118*State is *EnergyTransferStarted*

Main (Test scenario)

| Charging Station | CSMS |
|--|---|
| 2. The Charging Station responds with a SetChargingProfileResponse | 1. The Test System sends a SetChargingProfileRequest with chargingProfile.chargingProfilePurpose <i>TxProfile</i> chargingProfile.transactionId <i><Provided transactionId from before></i> chargingProfile.chargingSchedule[0].id <i><Id generated by Test System></i> chargingProfile.chargingSchedule[0].chargingRateUnit <i><Configured chargingRateUnit></i> chargingProfile.chargingSchedule[0].chargingSchedulePeriod[0].startPeriod <i>0</i> chargingProfile.chargingSchedule[0].chargingSchedulePeriod[0].limit <i>8 * <limit multiplier></i> <i>Note: Check Determine Charging Profile Limit Multiplier for <limit multiplier></i> |
| 3. The Charging Station sends a NotifyEVChargingScheduleRequest . <u>Note(s):</u> - EV should send its calculated charging profile, but Charging Station has no control over it. If EV does not send it, Charging Station is recommended to return a schedule matching the provided charging schedule from CSMS. | 4. The Test System responds with a NotifyEVChargingScheduleResponse . With status <i>Accepted</i> |

| Tool validations |
|--|
| <p>* Step 2: (Message: SetChargingProfileResponse) status <i>Accepted</i></p> <p>* Step 3: <i>Check that EV charging schedule matches or fits within charging profile</i> (Message: NotifyEVChargingScheduleRequest) evseld <i><Configured evseld></i> chargingSchedule.chargingSchedule[0].chargingRateUnit <i><Configured chargingRateUnit></i> chargingSchedule.chargingSchedule[0].chargingSchedulePeriod[0].startPeriod <i>0</i> chargingProfile.chargingSchedule[0].chargingSchedulePeriod[0].limit <i><= 8 * <limit multiplier></i> <i>Note: Check Determine Charging Profile Limit Multiplier for <limit multiplier></i></p> |
| <p>Post scenario validations: N/a</p> |

L Firmware Management

TC_L_01_CS: Secure Firmware Update - Installation successful

| | |
|-------------------|---|
| Test case name | Secure Firmware Update - Installation successful |
| Test case Id | TC_L_01_CS |
| Use case Id(s) | L01 |
| Requirement(s) | L01.FR.01,L01.FR.04,L01.FR.05,L01.FR.09,L01.FR.10,L01.FR.12,L01.FR.13,L01.FR.15,L01.FR.20,L01.FR.21,L01.FR.23 |
| System under test | Charging Station |
| Description | The CSMS is able to request the Charging Station to securely download and install a new firmware by sending an UpdateFirmwareRequest with a signingCertificate. |
| Purpose | To verify if the Charging Station is able to securely download and install a new firmware. |
| Prerequisite(s) | A file server has been setup according to the (by the Charging Station) supported file transfer protocol(s), indicated by the configuration variable FileTransferProtocols . |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a UpdateFirmwareResponse | 1. The Test System sends a UpdateFirmwareRequest with firmware.installDateTime <Current DateTime - 2 hours> firmware.location <Configured firmware_location> firmware.retrieveDateTime <Current DateTime - 2 hours> firmware.signingCertificate <Configured signingCertificate> firmware.signature <Configured signature> |
| 3. The Charging Station sends a FirmwareStatusNotificationRequest With status <i>Downloading</i> | 4. The Test System responds with a FirmwareStatusNotificationResponse |
| 5. The Charging Station sends a FirmwareStatusNotificationRequest With status <i>Downloaded</i> | 6. The Test System responds with a FirmwareStatusNotificationResponse |
| 7. The Charging Station sends a FirmwareStatusNotificationRequest With status <i>SignatureVerified</i> | 8. The Test System responds with a FirmwareStatusNotificationResponse |
| 9. The Charging Station notifies the CSMS about the current state of all connectors. <u>Note(s):</u> - This step is optional. The Charging Station may wants to set its connectors to <i>Unavailable</i> , before proceeding installing the new firmware. | 10. The Test System responds accordingly. |
| 11. Execute Reusable State <i>RebootBeforeFirmwareInstallation</i> | |
| <u>Note:</u> This step only needs to be executed if the Charging Station needs to reboot before firmware <u>installation</u> . | |

| Main (Test scenario) | |
|---|---|
| <p>12. The Charging Station sends a FirmwareStatusNotificationRequest With status <i>Installing</i></p> <p><u>Note(s):</u> - This step only needs to be executed if the Charging Station did NOT reboot before firmware <u>installation</u>, at step 11.</p> | <p>13. The Test System responds with a FirmwareStatusNotificationResponse</p> |
| <p>14. Execute Reusable State <i>RebootBeforeFirmwareActivation</i></p> <p><u>Note:</u> This step only needs to be executed if the Charging Station needs to reboot before firmware <u>activation</u>.</p> | |
| <p>15. The Test System waits for the Charging Station to reconnect.</p> <p><u>Note:</u> This step only needs to be executed if the Charging Station did not reboot/reconnect up until this point. The Charging Station should at least reconnect to reestablish the protocol version handshake.</p> <p><u>Note:</u> Step 16 through 21 can be send in a different order.</p> | |
| <p>16. The Charging Station notifies the CSMS about the current state of all connectors.</p> <p><u>Note(s):</u> - This step only needs to be executed if the connectors were previously set to <i>Unavailable</i> (at step 9) and the Charging Station did not report setting them back to <i>Available</i> (after a reboot sequence at step 11 or 14) yet.</p> | <p>17. The Test System responds accordingly.</p> |
| <p>18. The Charging Station sends a FirmwareStatusNotificationRequest With status <i>Installed</i></p> | <p>19. The Test System responds with a FirmwareStatusNotificationResponse</p> |
| <p>20. The Charging Station sends a SecurityEventNotificationRequest With type <i>FirmwareUpdated</i></p> | <p>21. The Test System responds with a SecurityEventNotificationResponse</p> |

| Tool validations |
|---|
| <p>* Step 2: Message UpdateFirmwareResponse - status <i>Accepted</i></p> <p>* Step 3: Message FirmwareStatusNotificationRequest - status <i>Downloading</i></p> <p>* Step 5: Message FirmwareStatusNotificationRequest - status <i>Downloaded</i></p> <p>* Step 7: Message FirmwareStatusNotificationRequest - status <i>SignatureVerified</i></p> <p>* Step 9: Message: StatusNotificationRequest - connectorStatus <i>Unavailable</i></p> <p>Or</p> <p>Message: NotifyEventRequest - eventData[0].trigger <i>Delta</i> - eventData[0].actualValue <i>"Unavailable"</i> - eventData[0].component.name <i>"Connector"</i> - eventData[0].variable.name <i>"AvailabilityState"</i></p> <p>* Step 12: Message FirmwareStatusNotificationRequest - status <i>Installing</i></p> <p>* Step 16: Message: StatusNotificationRequest - connectorStatus <i>Available</i></p> <p>Or</p> <p>Message: NotifyEventRequest - eventData[0].trigger <i>Delta</i> - eventData[0].actualValue <i>"Available"</i> - eventData[0].component.name <i>"Connector"</i> - eventData[0].variable.name <i>"AvailabilityState"</i></p> <p>* Step 18: Message FirmwareStatusNotificationRequest - status <i>Installed</i></p> <p>* Step 20: Message SecurityEventNotificationRequest - type <i>FirmwareUpdated</i></p> |
| <p>Post scenario validations: N/a</p> |

TC_L_02_CS: Secure Firmware Update - InstallScheduled

| | |
|--------------------------|---|
| Test case name | Secure Firmware Update - InstallScheduled |
| Test case Id | TC_L_02_CS |
| Use case Id(s) | L01 |
| Requirement(s) | L01.FR.01,L01.FR.04,L01.FR.05,L01.FR.09,L01.FR.10,L01.FR.12,L01.FR.15,L01.FR.16,L01.FR.20,L01.FR.21,L01.FR.23 |
| System under test | Charging Station |
| Description | The CSMS is able to request the Charging Station to securely download and install a new firmware by sending an UpdateFirmwareRequest with a signingCertificate . |
| Purpose | To verify if the Charging Station is able securely download a new firmware and schedule its installation. |
| Prerequisite(s) | <ul style="list-style-type: none"> - A file server has been setup according to the (by the Charging Station) supported file transfer protocol(s), indicated by the configuration variable FileTransferProtocols. - The Test System configuration firmware installDateTime needs to set to a future dateTime. |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a UpdateFirmwareResponse | 1. The Test System sends a UpdateFirmwareRequest with firmware.location <Configured <i>firmware_location</i> > firmware.retrieveDateTime <Current <i>DateTime</i> - 2 hours> firmware.signingCertificate <Configured <i>signingCertificate</i> > firmware.signature <Configured <i>signature</i> > firmware.installDateTime <Current <i>DateTime</i> + <Configured <i>Install Offset Period</i> >> |
| 3. The Charging Station sends a FirmwareStatusNotificationRequest With status <i>Downloading</i> | 4. The Test System responds with a FirmwareStatusNotificationResponse |
| 5. The Charging Station sends a FirmwareStatusNotificationRequest With status <i>Downloaded</i> | 6. The Test System responds with a FirmwareStatusNotificationResponse |
| 7. The Charging Station sends a FirmwareStatusNotificationRequest With status <i>SignatureVerified</i> | 8. The Test System responds with a FirmwareStatusNotificationResponse |
| 9. The Charging Station sends a FirmwareStatusNotificationRequest With status <i>InstallScheduled</i> | 10. The Test System responds with a FirmwareStatusNotificationResponse |
| <u>Note(s):</u> - The Charging Station will start installing the firmware after the set installDateTime is reached. | |
| 11. The Charging Station notifies the CSMS about the current state of all connectors. | 12. The Test System responds accordingly. |
| <u>Note(s):</u> - This step is optional. The Charging Station may wants to set its connectors to <i>Unavailable</i> , before proceeding installing the new firmware. | |

| | |
|---|--|
| Main (Test scenario) | |
| 13. Execute Reusable State <i>RebootBeforeFirmwareInstallation</i> | |
| <u>Note:</u> This step only needs to be executed if the Charging Station needs to reboot before firmware <u>installation</u> . | |
| 14. The Charging Station sends a FirmwareStatusNotificationRequest With status <i>Installing</i> | 15. The Test System responds with a FirmwareStatusNotificationResponse |
| <u>Note(s):</u> - This step only needs to be executed if the Charging Station did NOT reboot before firmware <u>installation</u> , at step 13. | |
| 16. Execute Reusable State <i>RebootBeforeFirmwareActivation</i> | |
| <u>Note:</u> This step only needs to be executed if the Charging Station needs to reboot before firmware <u>activation</u> . | |
| 17. The Test System waits for the Charging Station to reconnect. | |
| <u>Note:</u> This step only needs to be executed if the Charging Station did not reboot/reconnect up until this point. The Charging Station should at least reconnect to reestablish the protocol version handshake. | |
| <u>Note:</u> Step 18 through 23 can be send in a different order. | |
| 18. The Charging Station notifies the CSMS about the current state of all connectors. | 19. The Test System responds accordingly. |
| <u>Note(s):</u> - This step only needs to be executed if the connectors were previously set to <i>Unavailable</i> (at step 11) and the Charging Station did not report setting them back to <i>Available</i> (after a reboot sequence at step 13 or 16) yet. | |
| 20. The Charging Station sends a FirmwareStatusNotificationRequest With status <i>Installed</i> | 21. The Test System responds with a FirmwareStatusNotificationResponse |
| 22. The Charging Station sends a SecurityEventNotificationRequest With type <i>FirmwareUpdated</i> | 23. The Test System responds with a SecurityEventNotificationResponse |

| Tool validations |
|--|
| <p>* Step 2: Message UpdateFirmwareResponse - status <i>Accepted</i></p> <p>* Step 3: Message FirmwareStatusNotificationRequest - status <i>Downloading</i></p> <p>* Step 5: Message FirmwareStatusNotificationRequest - status <i>Downloaded</i></p> <p>* Step 7: Message FirmwareStatusNotificationRequest - status <i>SignatureVerified</i></p> <p>* Step 9: Message FirmwareStatusNotificationRequest - status <i>InstallScheduled</i></p> <p>* Step 11: Message: StatusNotificationRequest - connectorStatus <i>Unavailable</i></p> <p>Or Message: NotifyEventRequest - eventData[0].trigger <i>Delta</i> - eventData[0].actualValue <i>"Unavailable"</i> - eventData[0].component.name <i>"Connector"</i> - eventData[0].variable.name <i>"AvailabilityState"</i></p> <p>* Step 14: Message FirmwareStatusNotificationRequest - status <i>Installing</i></p> <p>* Step 18: Message: StatusNotificationRequest - connectorStatus <i>Available</i></p> <p>Or Message: NotifyEventRequest - eventData[0].trigger <i>Delta</i> - eventData[0].actualValue <i>"Available"</i> - eventData[0].component.name <i>"Connector"</i> - eventData[0].variable.name <i>"AvailabilityState"</i></p> <p>* Step 20: Message FirmwareStatusNotificationRequest - status <i>Installed</i></p> <p>* Step 22: Message SecurityEventNotificationRequest - type <i>FirmwareUpdated</i></p> |
| <p>Post scenario validations: N/a</p> |

TC_L_03_CS: Secure Firmware Update - DownloadScheduled

| | |
|-------------------|---|
| Test case name | Secure Firmware Update - DownloadScheduled |
| Test case Id | TC_L_03_CS |
| Use case Id(s) | L01 |
| Requirement(s) | L01.FR.01,L01.FR.04,L01.FR.05,L01.FR.09,L01.FR.10,L01.FR.12,L01.FR.13,L01.FR.15,L01.FR.20,L01.FR.21,L01.FR.23 |
| System under test | Charging Station |
| Description | The CSMS is able to request the Charging Station to securely download and install a new firmware by sending an UpdateFirmwareRequest with a signingCertificate. |
| Purpose | To verify if the Charging Station is able to schedule securely downloading a new firmware. |
| Prerequisite(s) | - A file server has been setup according to the (by the Charging Station) supported file transfer protocol(s), indicated by the configuration variable FileTransferProtocols . - The Test System configuration firmware retrieveDateTime needs to set to a future dateTime. |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State(s):

N/a

Main (Test scenario)

| Charging Station | CSMS |
|--|---|
| 2. The Charging Station responds with a UpdateFirmwareResponse | 1. The Test System sends a UpdateFirmwareRequest with firmware.installDateTime <Current DateTime - 2 hours> firmware.location <Configured firmware_location> firmware.retrieveDateTime <Current DateTime + <Configured Download Offset Period>> firmware.signingCertificate <Configured signingCertificate> firmware.signature <Configured signature> |
| 3. The Charging Station sends a FirmwareStatusNotificationRequest With status DownloadScheduled <u>Note(s):</u> - The Charging Station will start downloading the firmware after the set retrieveDateTime is reached. | 4. The Test System responds with a FirmwareStatusNotificationResponse |
| 5. The Charging Station sends a FirmwareStatusNotificationRequest With status Downloading | 6. The Test System responds with a FirmwareStatusNotificationResponse |
| 7. The Charging Station sends a FirmwareStatusNotificationRequest With status Downloaded | 8. The Test System responds with a FirmwareStatusNotificationResponse |
| 9. The Charging Station sends a FirmwareStatusNotificationRequest With status SignatureVerified | 10. The Test System responds with a FirmwareStatusNotificationResponse |
| 11. The Charging Station notifies the CSMS about the current state of all connectors. <u>Note(s):</u> - This step is optional. The Charging Station may wants to set its connectors to Unavailable, before proceeding installing the new firmware. | 12. The Test System responds accordingly. |

| | |
|---|--|
| Main (Test scenario) | |
| 13. Execute Reusable State <i>RebootBeforeFirmwareInstallation</i> | |
| <u>Note:</u> This step only needs to be executed if the Charging Station needs to reboot before firmware <u>installation</u> . | |
| 14. The Charging Station sends a FirmwareStatusNotificationRequest With status <i>Installing</i> | 15. The Test System responds with a FirmwareStatusNotificationResponse |
| <u>Note(s):</u> - This step only needs to be executed if the Charging Station did NOT reboot before firmware <u>installation</u> , at step 13. | |
| 16. Execute Reusable State <i>RebootBeforeFirmwareActivation</i> | |
| <u>Note:</u> This step only needs to be executed if the Charging Station needs to reboot before firmware <u>activation</u> . | |
| 17. The Test System waits for the Charging Station to reconnect. | |
| <u>Note:</u> This step only needs to be executed if the Charging Station did not reboot/reconnect up until this point. The Charging Station should at least reconnect to reestablish the protocol version handshake. | |
| <u>Note:</u> Step 18 through 23 can be send in a different order. | |
| 18. The Charging Station notifies the CSMS about the current state of all connectors. | 19. The Test System responds accordingly. |
| <u>Note(s):</u> - This step only needs to be executed if the connectors were previously set to <i>Unavailable</i> (at step 11) and the Charging Station did not report setting them back to <i>Available</i> (after a reboot sequence at step 13 or 16) yet. | |
| 20. The Charging Station sends a FirmwareStatusNotificationRequest With status <i>Installed</i> | 21. The Test System responds with a FirmwareStatusNotificationResponse |
| 22. The Charging Station sends a SecurityEventNotificationRequest With type <i>FirmwareUpdated</i> | 23. The Test System responds with a SecurityEventNotificationResponse |

| Tool validations |
|---|
| <p>* Step 2: Message UpdateFirmwareResponse - status <i>Accepted</i></p> <p>* Step 3: Message FirmwareStatusNotificationRequest - status <i>DownloadScheduled</i></p> <p>* Step 5: Message FirmwareStatusNotificationRequest - status <i>Downloading</i></p> <p>* Step 7: Message FirmwareStatusNotificationRequest - status <i>Downloaded</i></p> <p>* Step 9: Message FirmwareStatusNotificationRequest - status <i>SignatureVerified</i></p> <p>* Step 11: Message: StatusNotificationRequest - connectorStatus <i>Unavailable</i></p> <p>Or</p> <p>Message: NotifyEventRequest - eventData[0].trigger <i>Delta</i> - eventData[0].actualValue <i>"Unavailable"</i> - eventData[0].component.name <i>"Connector"</i> - eventData[0].variable.name <i>"AvailabilityState"</i></p> <p>* Step 14: Message FirmwareStatusNotificationRequest - status <i>Installing</i></p> <p>* Step 18: Message: StatusNotificationRequest - connectorStatus <i>Available</i></p> <p>Or</p> <p>Message: NotifyEventRequest - eventData[0].trigger <i>Delta</i> - eventData[0].actualValue <i>"Available"</i> - eventData[0].component.name <i>"Connector"</i> - eventData[0].variable.name <i>"AvailabilityState"</i></p> <p>* Step 20: Message FirmwareStatusNotificationRequest - status <i>Installed</i></p> <p>* Step 22: Message SecurityEventNotificationRequest - type <i>FirmwareUpdated</i></p> |
| <p>Post scenario validations: N/a</p> |

TC_L_05_CS: Secure Firmware Update - InvalidCertificate

| | |
|-------------------|---|
| Test case name | Secure Firmware Update - InvalidCertificate |
| Test case Id | TC_L_05_CS |
| Use case Id(s) | L01 |
| Requirement(s) | L01.FR.01,L01.FR.02,L01.FR.10,L01.FR.20,L01.FR.21,L01.FR.22 |
| System under test | Charging Station |
| Description | The CSMS is able to request the Charging Station to securely download and install a new firmware by sending an UpdateFirmwareRequest with a signingCertificate. |
| Purpose | To verify if the Charging Station is able to identify it receiving an invalid signing certificate and report this to the CSMS. |
| Prerequisite(s) | N/a |

| |
|--|
| Before (Preparations) |
| Configuration State: <Generated Invalid Firmware SigningCertificate> should be a trusted certificate and not be the same as the <Configured Valid Firmware SigningCertificate> |
| Memory State: N/a |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|---|---|
| Charging Station | CSMS |
| 2. The Charging Station responds with a UpdateFirmwareResponse | 1. The Test System sends a UpdateFirmwareRequest with firmware.installDateTime <Current DateTime - 2 hours> firmware.location <Configured firmware_location> firmware.retrieveDateTime <Current DateTime - 2 hours> firmware.signingCertificate <Generated invalid firmware signingCertificate> firmware.signature <Configured signature> |
| 3. The Charging Station sends a SecurityEventNotificationRequest . With type InvalidFirmwareSigningCertificate | 4. The Test System responds with a SecurityEventNotificationResponse . |

| |
|---|
| Tool validations |
| * Step 2: Message UpdateFirmwareResponse - status InvalidCertificate OR RevokedCertificate * Step 3: Message SecurityEventNotificationRequest - type InvalidFirmwareSigningCertificate |
| Post scenario validations: N/a |

TC_L_06_CS: Secure Firmware Update - InvalidSignature

| | |
|--------------------------|---|
| Test case name | Secure Firmware Update - InvalidSignature |
| Test case Id | TC_L_06_CS |
| Use case Id(s) | L01 |
| Requirement(s) | L01.FR.01,L01.FR.03,L01.FR.04,L01.FR.10,L01.FR.20 |
| System under test | Charging Station |
| Description | The CSMS is able to request the Charging Station to securely download and install a new firmware by sending an UpdateFirmwareRequest with a signingCertificate . |
| Purpose | To verify if the Charging Station is able to identify if the signature is invalid and report this to the CSMS. |
| Prerequisite(s) | A file server has been setup according to the (by the Charging Station) supported file transfer protocol(s), indicated by the configuration variable FileTransferProtocols . |

| |
|---|
| Before (Preparations) |
| Configuration State: <Configured invalid firmware signature> should be a real signature |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a UpdateFirmwareResponse | 1. The Test System sends a UpdateFirmwareRequest with firmware.installDateTime <Current DateTime - 2 hours> firmware.location <Configured firmware_location> firmware.retrieveDateTime <Current DateTime - 2 hours> firmware.signingCertificate <Configured signingCertificate> firmware.signature <Configured invalid firmware signature> |
| 3. The Charging Station sends a FirmwareStatusNotificationRequest . With status <i>Downloading</i> | 4. The Test System responds with a FirmwareStatusNotificationResponse . |
| 5. The Charging Station sends a FirmwareStatusNotificationRequest . With status <i>Downloaded</i> | 6. The Test System responds with a FirmwareStatusNotificationResponse . |
| <u>Note:</u> Step 7 through 10 can be sent in a different order. | |
| 7. The Charging Station sends a FirmwareStatusNotificationRequest . With status <i>InvalidSignature</i> | 8. The Test System responds with a FirmwareStatusNotificationResponse . |
| 9. The Charging Station sends a SecurityEventNotificationRequest . With type <i>InvalidFirmwareSignature</i> | 10. The Test System responds with a SecurityEventNotificationResponse . |

| Tool validations |
|--|
| <div>* Step 2: Message UpdateFirmwareResponse - status <i>Accepted</i></div> <div>* Step 3: Message FirmwareStatusNotificationRequest - status <i>Downloading</i></div> <div>* Step 5: Message FirmwareStatusNotificationRequest - status <i>Downloaded</i></div> <div>* Step 7: Message FirmwareStatusNotificationRequest - status <i>InvalidSignature</i></div> <div>* Step 9: Message SecurityEventNotificationRequest - type <i>InvalidFirmwareSignature</i></div> |
| <div>Post scenario validations: N/a</div> |

TC_L_07_CS: Secure Firmware Update - DownloadFailed

| | |
|-------------------|---|
| Test case name | Secure Firmware Update - DownloadFailed |
| Test case Id | TC_L_07_CS |
| Use case Id(s) | L01 |
| Requirement(s) | L01.FR.01,L01.FR.10,L01.FR.20 |
| System under test | Charging Station |
| Description | The CSMS is able to request the Charging Station to securely download and install a new firmware by sending an UpdateFirmwareRequest with a signingCertificate. |
| Purpose | To verify if the Charging Station is able to report to the CSMS when it is unable to download the new firmware. |
| Prerequisite(s) | - A file server has been setup according to the (by the Charging Station) supported file transfer protocol(s), indicated by the configuration variable FileTransferProtocols . - The at the Test System configured invalid firmware location needs to point to a not existing firmware file name. |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State(s):

N/a

Main (Test scenario)

| Charging Station | CSMS |
|--|--|
| 2. The Charging Station responds with a UpdateFirmwareResponse | 1. The Test System sends a UpdateFirmwareRequest with firmware.installDateTime <Current DateTime - 2 hours> firmware.location <Configured firmware location> + "_does_not_exist" firmware.retrieveDateTime _<Current DateTime - 2 hours> firmware.signingCertificate <Configured signingCertificate> firmware.signature <Configured signature> |
| 3. The Charging Station sends a FirmwareStatusNotificationRequest . With status <i>Downloading</i> <u>Note(s):</u> - This step is optional. The Charging Station may immediately identify downloading the firmware is not possible. | 4. The Test System responds with a FirmwareStatusNotificationResponse . |
| 5. The Charging Station sends a FirmwareStatusNotificationRequest . With status <i>DownloadFailed</i> | 6. The Test System responds with a FirmwareStatusNotificationResponse . |

Tool validations

* Step 2:

Message **UpdateFirmwareResponse**- **status** *Accepted*

* Step 3:

Message **FirmwareStatusNotificationRequest**- **status** *Downloading*

* Step 5:

Message **FirmwareStatusNotificationRequest**- **status** *DownloadFailed*

| |
|--|
| Tool validations |
| Post scenario validations: N/a |

TC_L_08_CS: Secure Firmware Update - InstallVerificationFailed or InstallationFailed

| | |
|--------------------------|---|
| Test case name | Secure Firmware Update - InstallVerificationFailed or InstallationFailed |
| Test case Id | TC_L_08_CS |
| Use case Id(s) | L01 |
| Requirement(s) | L01.FR.01,L01.FR.10,L01.FR.12,L01.FR.20 |
| System under test | Charging Station |
| Description | The CSMS is able to request the Charging Station to securely download and install a new firmware by sending an UpdateFirmwareRequest with a signingCertificate. |
| Purpose | To verify if the Charging Station is able to report to the CSMS when the firmware verification fails. |
| Prerequisite(s) | <ul style="list-style-type: none"> - A file server has been setup according to the (by the Charging Station) supported file transfer protocol(s), indicated by the configuration variable FileTransferProtocols. - The at the Test System configured invalid firmware location needs to point to a firmware file that causes an InstallVerificationFailed. |

| |
|---|
| Before (Preparations) |
| Configuration State: <Configured invalid firmware location> should point to existing firmware that causes an InstallVerificationFailed <Configured invalid firmware signingCertificate> should be a trusted signingCertificate <Configured invalid firmware signature> should be a real signature |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a UpdateFirmwareResponse | 1. The Test System sends a UpdateFirmwareRequest with firmware.installDateTime <Current DateTime - 2 hours> firmware.location <Configured invalid firmware location> firmware.retrieveDateTime <Current DateTime + <Current DateTime - 2 hours>> firmware.signingCertificate <Configured invalid firmware signingCertificate> firmware.signature <Configured invalid firmware signature> |
| 3. The Charging Station sends a FirmwareStatusNotificationRequest With status Downloading | 4. The Test System responds with a FirmwareStatusNotificationResponse |
| 5. The Charging Station sends a FirmwareStatusNotificationRequest With status Downloaded | 6. The Test System responds with a FirmwareStatusNotificationResponse |
| 7. The Charging Station sends a FirmwareStatusNotificationRequest With status SignatureVerified | 8. The Test System responds with a FirmwareStatusNotificationResponse |
| 9. The Charging Station notifies the CSMS about the current state of all connectors. | 10. The Test System responds accordingly. |
| Note(s): - This step is optional. The Charging Station may wants to set its connectors to Unavailable, before proceeding installing the new firmware. | |

| Main (Test scenario) | |
|--|--|
| 11. Execute Reusable State <i>RebootBeforeFirmwareInstallation</i> | |
| <u>Note:</u> <i>This step only needs to be executed if the Charging Station needs to reboot before firmware <u>installation</u>.</i> | |
| 12. The Charging Station sends a FirmwareStatusNotificationRequest With status <i>Installing</i> | 13. The Test System responds with a FirmwareStatusNotificationResponse |
| <u>Note(s):</u> <i>- This step only needs to be executed if the Charging Station did NOT reboot before firmware <u>installation</u>, at step 11.</i> | |
| <u>Note:</u> <i>Step 14 through 17 can be send in a different order.</i> | |
| 14. The Charging Station notifies the CSMS about the current state of all connectors. | 15. The Test System responds accordingly. |
| <u>Note(s):</u> <i>- This step only needs to be executed if the connectors were previously set to Unavailable (at step 9) and the Charging Station did not report setting them back to Available (after the reboot sequence at step 11) yet.</i> <i>- And if the Charging Station did not become inoperative after the firmware update failure. It is recommended for a Charging Station to fallback to the previous firmware after a firmware update failure.</i> | |
| 16. The Charging Station sends a FirmwareStatusNotificationRequest With status <i>InstallVerificationFailed</i> or <i>InstallationFailed</i> | 17. The Test System responds with a FirmwareStatusNotificationResponse |

| Tool validations |
|---|
| <p>* Step 2: Message UpdateFirmwareResponse - status <i>Accepted</i></p> <p>* Step 3: Message FirmwareStatusNotificationRequest - status <i>Downloading</i></p> <p>* Step 5: Message FirmwareStatusNotificationRequest - status <i>Downloaded</i></p> <p>* Step 7: Message FirmwareStatusNotificationRequest - status <i>SignatureVerified</i></p> <p>* Step 9: Message: StatusNotificationRequest - connectorStatus <i>Unavailable</i></p> <p>Or</p> <p>Message: NotifyEventRequest - eventData[0].trigger <i>Delta</i> - eventData[0].actualValue <i>"Unavailable"</i> - eventData[0].component.name <i>"Connector"</i> - eventData[0].variable.name <i>"AvailabilityState"</i></p> <p>* Step 12: Message FirmwareStatusNotificationRequest - status <i>Installing</i></p> <p>* Step 14: Message: StatusNotificationRequest - connectorStatus <i>Available</i></p> <p>Or</p> <p>Message: NotifyEventRequest - eventData[0].trigger <i>Delta</i> - eventData[0].actualValue <i>"Available"</i> - eventData[0].component.name <i>"Connector"</i> - eventData[0].variable.name <i>"AvailabilityState"</i></p> <p>* Step 16: Message FirmwareStatusNotificationRequest - status <i>InstallVerificationFailed</i> or <i>InstallationFailed</i></p> |
| <p>Post scenario validations: N/a</p> |

TC_L_10_CS: Secure Firmware Update - AcceptedCanceled

| | |
|-------------------|---|
| Test case name | Secure Firmware Update - AcceptedCanceled |
| Test case Id | TC_L_10_CS |
| Use case Id(s) | L01 |
| Requirement(s) | L01.FR.01,L01.FR.10,L01.FR.20,L01.FR.24 |
| System under test | Charging Station |
| Description | The CSMS is able to request the Charging Station to securely download and install a new firmware by sending an UpdateFirmwareRequest with a signingCertificate. |
| Purpose | To verify if the Charging Station is able to cancel an ongoing firmware update and start a new one, when receiving an UpdateFirmwareRequest from the CSMS. |
| Prerequisite(s) | <ul style="list-style-type: none"> - A file server has been setup according to the (by the Charging Station) supported file transfer protocol(s), indicated by the configuration variable FileTransferProtocols. - The Charging Station is able to cancel an ongoing firmware update while it is busy downloading a new firmware file. |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State(s):

N/a

Main (Test scenario)

| Charging Station | CSMS |
|---|--|
| 2. The Charging Station responds with a UpdateFirmwareResponse With status <i>Accepted</i> | 1. The Test System sends a UpdateFirmwareRequest with requestId = <#1> firmware.installDateTime <Current DateTime - 2 hours> firmware.location <Configured firmware_location> firmware.retrieveDateTime <Current DateTime - 2 hours> firmware.signingCertificate <Configured signingCertificate> firmware.signature <Configured signature> |
| 3. The Charging Station sends a FirmwareStatusNotificationRequest With requestId <#1> and status <i>Downloading</i> | 4. The Test System responds with a FirmwareStatusNotificationResponse |
| 6. The Charging Station responds with a UpdateFirmwareResponse With requestId <#1> and status <i>AcceptedCanceled</i> | 5. The Test System sends a UpdateFirmwareRequest with requestId = <#2> firmware.installDateTime <Current DateTime - 2 hours> firmware.location <Configured firmware_location> firmware.retrieveDateTime <Current DateTime - 2 hours> firmware.signingCertificate <Configured signingCertificate> firmware.signature <Configured signature> |
| 7. The Charging Station sends a FirmwareStatusNotificationRequest With requestId <#2> and status <i>Downloading</i> | 8. The Test System responds with a FirmwareStatusNotificationResponse |
| 9. The Charging Station sends a FirmwareStatusNotificationRequest With status <i>Downloaded</i> | 10. The Test System responds with a FirmwareStatusNotificationResponse |
| 11. The Charging Station sends a FirmwareStatusNotificationRequest With status <i>SignatureVerified</i> | 12. The Test System responds with a FirmwareStatusNotificationResponse |

| | |
|---|---|
| Main (Test scenario) | |
| <p>13. The Charging Station notifies the CSMS about the current state of all connectors.</p> <p><u>Note(s):</u> - This step is optional. The Charging Station may want to set its connectors to Unavailable, before proceeding installing the new firmware.</p> | <p>14. The Test System responds accordingly.</p> |
| <p>15. Execute Reusable State <i>RebootBeforeFirmwareInstallation</i></p> <p><u>Note:</u> This step only needs to be executed if the Charging Station needs to reboot before firmware <u>installation</u>.</p> | |
| <p>16. The Charging Station sends a FirmwareStatusNotificationRequest With status <i>Installing</i></p> <p><u>Note(s):</u> - This step only needs to be executed if the Charging Station did NOT reboot before firmware <u>installation</u>, at step 15.</p> | <p>17. The Test System responds with a FirmwareStatusNotificationResponse</p> |
| <p>18. Execute Reusable State <i>RebootBeforeFirmwareActivation</i></p> <p><u>Note:</u> This step only needs to be executed if the Charging Station needs to reboot before firmware <u>activation</u>.</p> | |
| <p>19. The Test System waits for the Charging Station to reconnect.</p> <p><u>Note:</u> This step only needs to be executed if the Charging Station did not reboot/reconnect up until this point. The Charging Station should at least reconnect to reestablish the protocol version handshake.</p> <p><u>Note:</u> Step 20 through 25 can be send in a different order.</p> | |
| <p>20. The Charging Station notifies the CSMS about the current state of all connectors.</p> <p><u>Note(s):</u> - This step only needs to be executed if the connectors were previously set to Unavailable (at step 13) and the Charging Station did not report setting them back to Available (after a reboot sequence at step 15 or 18) yet.</p> | <p>21. The Test System responds accordingly.</p> |
| <p>22. The Charging Station sends a FirmwareStatusNotificationRequest With status <i>Installed</i></p> | <p>23. The Test System responds with a FirmwareStatusNotificationResponse</p> |
| <p>24. The Charging Station sends a SecurityEventNotificationRequest With type <i>FirmwareUpdated</i></p> | <p>25. The Test System responds with a SecurityEventNotificationResponse</p> |

| Tool validations |
|--|
| <p>* Step 2: Message UpdateFirmwareResponse - status <i>Accepted</i></p> <p>* Step 3: Message FirmwareStatusNotificationRequest - status <i>Downloading</i> - requestId = <#1></p> <p>* Step 6: Message UpdateFirmwareResponse - status <i>AcceptedCanceled</i> A FirmwareStatusNotificationRequest <i>DownloadFailed</i> or <i>InstallationFailed</i> may be sent for requestId <#1> before or after step 6.</p> <p>(The requestId at the FirmwareStatusNotificationRequest messages must refer to the id <#2> from the second UpdateFirmwareRequest from this point on)</p> <p>* Step 7: Message FirmwareStatusNotificationRequest - status <i>Downloading</i></p> <p>* Step 9: Message FirmwareStatusNotificationRequest - status <i>Downloaded</i></p> <p>* Step 11: Message FirmwareStatusNotificationRequest - status <i>SignatureVerified</i></p> <p>* Step 13: Message: StatusNotificationRequest - connectorStatus <i>Unavailable</i></p> <p>Or Message: NotifyEventRequest - eventData[0].trigger <i>Delta</i> - eventData[0].actualValue <i>"Unavailable"</i> - eventData[0].component.name <i>"Connector"</i> - eventData[0].variable.name <i>"AvailabilityState"</i></p> <p>* Step 16: Message FirmwareStatusNotificationRequest - status <i>Installing</i></p> <p>* Step 20: Message: StatusNotificationRequest - connectorStatus <i>Available</i></p> <p>Or Message: NotifyEventRequest - eventData[0].trigger <i>Delta</i> - eventData[0].actualValue <i>"Available"</i> - eventData[0].component.name <i>"Connector"</i> - eventData[0].variable.name <i>"AvailabilityState"</i></p> <p>* Step 22: Message FirmwareStatusNotificationRequest - status <i>Installed</i></p> <p>* Step 24: Message SecurityEventNotificationRequest - type <i>FirmwareUpdated</i></p> |
| <p>Post scenario validations: N/a</p> |

TC_L_11_CS: Secure Firmware Update - Unable to cancel

| | |
|-------------------|--|
| Test case name | Secure Firmware Update - Unable to cancel |
| Test case Id | TC_L_11_CS |
| Use case Id(s) | L01 |
| Requirement(s) | L01.FR.01,L01.FR.10,L01.FR.20,L01.FR.27 |
| System under test | Charging Station |
| Description | The CSMS is able to request the Charging Station to securely download and install a new firmware by sending an <code>UpdateFirmwareRequest</code> with a <code>signingCertificate</code> . |
| Purpose | To verify if the Charging Station is able to reject a firmware update request when it is unable to cancel an ongoing firmware update. |
| Prerequisite(s) | <ul style="list-style-type: none"> - A file server has been setup according to the (by the Charging Station) supported file transfer protocol(s), indicated by the configuration variable FileTransferProtocols. - The Charging Station is NOT able to cancel an ongoing firmware update. |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State(s):

N/a

Main (Test scenario)

| Charging Station | CSMS |
|--|---|
| 2. The Charging Station responds with a UpdateFirmwareResponse With status <i>Accepted</i> | 1. The Test System sends a UpdateFirmwareRequest with firmware.installDateTime <Current DateTime - 2 hours> firmware.location <Configured firmware_location> firmware.retrieveDateTime <Current DateTime - 2 hours> firmware.signingCertificate <Configured signingCertificate> firmware.signature <Configured signature> |
| 3. The Charging Station sends a FirmwareStatusNotificationRequest With status <i>Downloading</i> | 4. The Test System responds with a FirmwareStatusNotificationResponse |
| 6. The Charging Station responds with a UpdateFirmwareResponse With status <i>Rejected</i> | 5. The Test System sends a UpdateFirmwareRequest with firmware.installDateTime <Current DateTime - 2 hours> firmware.location <Configured firmware_location> firmware.retrieveDateTime <Current DateTime - 2 hours> firmware.signingCertificate <Configured signingCertificate> firmware.signature <Configured signature> |
| 7. The Charging Station sends a FirmwareStatusNotificationRequest With status <i>Downloaded</i> | 8. The Test System responds with a FirmwareStatusNotificationResponse |
| 9. The Charging Station sends a FirmwareStatusNotificationRequest With status <i>SignatureVerified</i> | 10. The Test System responds with a FirmwareStatusNotificationResponse |
| 11. The Charging Station notifies the CSMS about the current state of all connectors. <u>Note(s):</u> - This step is optional. The Charging Station may wants to set its connectors to <i>Unavailable</i> , before proceeding installing the new firmware. | 12. The Test System responds accordingly. |

| | |
|--|--|
| Main (Test scenario) | |
| 13. Execute Reusable State <i>RebootBeforeFirmwareInstallation</i> | |
| <u>Note:</u> <i>This step only needs to be executed if the Charging Station needs to reboot before firmware <u>installation</u>.</i> | |
| 14. The Charging Station sends a FirmwareStatusNotificationRequest With status <i>Installing</i> | 15. The Test System responds with a FirmwareStatusNotificationResponse |
| <u>Note(s):</u> <i>- This step only needs to be executed if the Charging Station did NOT reboot before firmware <u>installation</u>, at step 13.</i> | |
| 16. Execute Reusable State <i>RebootBeforeFirmwareActivation</i> | |
| <u>Note:</u> <i>This step only needs to be executed if the Charging Station needs to reboot before firmware <u>activation</u>.</i> | |
| 17. The Test System waits for the Charging Station to reconnect. | |
| <u>Note:</u> <i>This step only needs to be executed if the Charging Station did not reboot/reconnect up until this point. The Charging Station should at least reconnect to reestablish the protocol version handshake.</i> | |
| <u>Note:</u> <i>Step 18 through 23 can be send in a different order.</i> | |
| 18. The Charging Station notifies the CSMS about the current state of all connectors. | 19. The Test System responds accordingly. |
| <u>Note(s):</u> <i>- This step only needs to be executed if the connectors were previously set to Unavailable (at step 11) and the Charging Station did not report setting them back to Available (after a reboot sequence at step 13 or 16) yet.</i> | |
| 20. The Charging Station sends a FirmwareStatusNotificationRequest With status <i>Installed</i> | 21. The Test System responds with a FirmwareStatusNotificationResponse |
| 22. The Charging Station sends a SecurityEventNotificationRequest With type <i>FirmwareUpdated</i> | 23. The Test System responds with a SecurityEventNotificationResponse |

| Tool validations |
|---|
| <p>* Step 2: Message UpdateFirmwareResponse - status <i>Accepted</i></p> <p>* Step 3: Message FirmwareStatusNotificationRequest - status <i>Downloading</i></p> <p>* Step 6: Message UpdateFirmwareResponse - status <i>Rejected</i></p> <p>* Step 7: Message FirmwareStatusNotificationRequest - status <i>Downloaded</i></p> <p>* Step 9: Message FirmwareStatusNotificationRequest - status <i>SignatureVerified</i></p> <p>* Step 11: Message: StatusNotificationRequest - connectorStatus <i>Unavailable</i></p> <p>Or Message: NotifyEventRequest - eventData[0].trigger <i>Delta</i> - eventData[0].actualValue <i>"Unavailable"</i> - eventData[0].component.name <i>"Connector"</i> - eventData[0].variable.name <i>"AvailabilityState"</i></p> <p>* Step 14: Message FirmwareStatusNotificationRequest - status <i>Installing</i></p> <p>* Step 18: Message: StatusNotificationRequest - connectorStatus <i>Available</i></p> <p>Or Message: NotifyEventRequest - eventData[0].trigger <i>Delta</i> - eventData[0].actualValue <i>"Available"</i> - eventData[0].component.name <i>"Connector"</i> - eventData[0].variable.name <i>"AvailabilityState"</i></p> <p>* Step 20: Message FirmwareStatusNotificationRequest - status <i>Installed</i></p> <p>* Step 22: Message SecurityEventNotificationRequest - type <i>FirmwareUpdated</i></p> |
| <p>Post scenario validations: N/a</p> |

TC_L_12_CS: Secure Firmware Update - Unable to download/install firmware with ongoing transaction - AllowNewSessionsPendingFirmwareUpdate is true

| | |
|-------------------|--|
| Test case name | Secure Firmware Update - Unable to download/install firmware with ongoing transaction - AllowNewSessionsPendingFirmwareUpdate is true |
| Test case Id | TC_L_12_CS |
| Use case Id(s) | L01 |
| Requirement(s) | L01.FR.01,L01.FR.06,L01.FR.07,L01.FR.10,L01.FR.20 |
| System under test | Charging Station |
| Description | The CSMS is able to request the Charging Station to securely download and install a new firmware by sending an UpdateFirmwareRequest with a signingCertificate. |
| Purpose | To verify if the Charging Station is able to keep allowing new transactions when requested to update the firmware, while there is an ongoing transaction. |
| Prerequisite(s) | <ul style="list-style-type: none"> - A file server has been setup according to the (by the Charging Station) supported file transfer protocol(s), indicated by the configuration variable FileTransferProtocols. - The Charging Station is able to start more than one transaction at a time. - The Charging Station is unable to download AND install firmware while there is an ongoing transaction. |

| |
|---|
| Before (Preparations) |
| Configuration State: AllowNewSessionsPendingFirmwareUpdate is true (If implemented) |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> for <Configured connectorId> |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a UpdateFirmwareResponse With status Accepted | 1. The Test System sends a UpdateFirmwareRequest with firmware.installDateTime <Current DateTime - 2 hours> firmware.location <Configured firmware_location> firmware.retrieveDateTime <Current DateTime - 2 hours> firmware.signingCertificate <Configured signingCertificate> firmware.signature <Configured signature> |
| 3. The Charging Station sends a FirmwareStatusNotificationRequest With status DownloadScheduled | 4. The Test System responds with a FirmwareStatusNotificationResponse |
| 5. Execute Reusable State <i>EnergyTransferStarted</i> for <Configured second Connector> | |
| Note(s): - It is allowed to start a second transaction while there is a scheduled firmware update. | |
| 6. Execute Reusable State <i>ParkingBayUnoccupied</i> for <Configured connectorId> | |
| Note(s): - The Charging Station will proceed to this end state. This will cause the transaction to stop. | |
| 7. Execute Reusable State <i>ParkingBayUnoccupied</i> for <Configured second Connector> | |
| Note(s): - The Charging Station will proceed to this end state. This will cause the transaction to stop. - The Charging Station will start the firmware update process the moment this second transaction ends or when all interactions with the EV Driver are done (So after the cable has been unplugged, if there is no parking bay sensor). | |

| Main (Test scenario) | |
|---|---|
| 8. The Charging Station sends a FirmwareStatusNotificationRequest With status <i>Downloading</i> | 9. The Test System responds with a FirmwareStatusNotificationResponse |
| 10. The Charging Station sends a FirmwareStatusNotificationRequest With status <i>Downloaded</i> | 11. The Test System responds with a FirmwareStatusNotificationResponse |
| 12. The Charging Station sends a FirmwareStatusNotificationRequest With status <i>SignatureVerified</i> | 13. The Test System responds with a FirmwareStatusNotificationResponse |
| 14. The Charging Station notifies the CSMS about the current state of all connectors. <u>Note(s):</u> - This step is optional. The Charging Station may wants to set its connectors to <i>Unavailable</i> , before proceeding installing the new firmware. | 15. The Test System responds accordingly. |
| 16. Execute Reusable State <i>RebootBeforeFirmwareInstallation</i> <u>Note:</u> This step only needs to be executed if the Charging Station needs to reboot before firmware <u>installation</u> . | |
| 17. The Charging Station sends a FirmwareStatusNotificationRequest With status <i>Installing</i> <u>Note(s):</u> - This step only needs to be executed if the Charging Station did NOT reboot before firmware <u>installation</u> , at step 16. | 18. The Test System responds with a FirmwareStatusNotificationResponse |
| 19. Execute Reusable State <i>RebootBeforeFirmwareActivation</i> <u>Note:</u> This step only needs to be executed if the Charging Station needs to reboot before firmware <u>activation</u> . | |
| 20. The Test System waits for the Charging Station to reconnect. <u>Note:</u> This step only needs to be executed if the Charging Station did not reboot/reconnect up until this point. The Charging Station should at least reconnect to reestablish the protocol version handshake. <u>Note:</u> Step 21 through 26 can be send in a different order. | |
| 21. The Charging Station notifies the CSMS about the current state of all connectors. <u>Note(s):</u> - This step only needs to be executed if the connectors were previously set to <i>Unavailable</i> (at step 14) and the Charging Station did not report setting them back to <i>Available</i> (after a reboot sequence at step 16 or 19) yet. | 22. The Test System responds accordingly. |
| 23. The Charging Station sends a FirmwareStatusNotificationRequest With status <i>Installed</i> | 24. The Test System responds with a FirmwareStatusNotificationResponse |
| 25. The Charging Station sends a SecurityEventNotificationRequest With type <i>FirmwareUpdated</i> | 26. The Test System responds with a SecurityEventNotificationResponse |

Tool validations

* Step 2:

Message **UpdateFirmwareResponse**

- **status** *Accepted*

* Step 3:

Message **FirmwareStatusNotificationRequest**

- **status** *DownloadScheduled*

* Step 8:

Message **FirmwareStatusNotificationRequest**

- **status** *Downloading*

* Step 10:

Message **FirmwareStatusNotificationRequest**

- **status** *Downloaded*

* Step 12:

Message **FirmwareStatusNotificationRequest**

- **status** *SignatureVerified*

* Step 14:

Message: **StatusNotificationRequest**

- **connectorStatus** *Unavailable*

Or

Message: **NotifyEventRequest**

- **eventData[0].trigger** *Delta*

- **eventData[0].actualValue** *"Unavailable"*

- **eventData[0].component.name** *"Connector"*

- **eventData[0].variable.name** *"AvailabilityState"*

* Step 17:

Message **FirmwareStatusNotificationRequest**

- **status** *Installing*

* Step 21:

Message: **StatusNotificationRequest**

- **connectorStatus** *Available*

Or

Message: **NotifyEventRequest**

- **eventData[0].trigger** *Delta*

- **eventData[0].actualValue** *"Available"*

- **eventData[0].component.name** *"Connector"*

- **eventData[0].variable.name** *"AvailabilityState"*

* Step 23:

Message **FirmwareStatusNotificationRequest**

- **status** *Installed*

* Step 25:

Message **SecurityEventNotificationRequest**

- **type** *FirmwareUpdated*

Post scenario validations:

N/a

TC_L_13_CS: Secure Firmware Update - Unable to download/install firmware with ongoing transaction - AllowNewSessionsPendingFirmwareUpdate is false

| | |
|-------------------|---|
| Test case name | Secure Firmware Update - Unable to download/install firmware with ongoing transaction - AllowNewSessionsPendingFirmwareUpdate is false |
| Test case Id | TC_L_13_CS |
| Use case Id(s) | L01 |
| Requirement(s) | L01.FR.01,L01.FR.06,L01.FR.07,L01.FR.10,L01.FR.20 |
| System under test | Charging Station |
| Description | The CSMS is able to request the Charging Station to securely download and install a new firmware by sending an UpdateFirmwareRequest with a signingCertificate. |
| Purpose | To verify if the Charging Station is able to set its available connectors to Unavailable when requested to update the firmware, while there is an ongoing transaction. |
| Prerequisite(s) | <ul style="list-style-type: none"> - A file server has been setup according to the (by the Charging Station) supported file transfer protocol(s), indicated by the configuration variable FileTransferProtocols. - The configuration variable AllowNewSessionsPendingFirmwareUpdate is implemented. - The Charging Station is unable to download AND install firmware while there is an ongoing transaction. |

| |
|---|
| Before (Preparations) |
| Configuration State: AllowNewSessionsPendingFirmwareUpdate is <i>false</i> |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a UpdateFirmwareResponse With status <i>Accepted</i> | 1. The Test System sends a UpdateFirmwareRequest with firmware.installDateTime <Current DateTime - 2 hours> firmware.location <Configured firmware_location> firmware.retrieveDateTime <Current DateTime - 2 hours> firmware.signingCertificate <Configured signingCertificate> firmware.signature <Configured signature> |
| 3. The Charging Station sends a FirmwareStatusNotificationRequest With status <i>DownloadScheduled</i> <u>Note:</u> This step is optional. Part 2 specification only describes that this status needs to be send in case the retrieveDateTime is in the future. However it is also allowed to send this status if the Charging Station schedules the firmware download, because of an ongoing transaction. | 4. The Test System responds with a FirmwareStatusNotificationResponse |
| 5. The Charging Station notifies the CSMS about the current state of its Available connector(s). <u>Note(s):</u> - This step needs to be executed for all connectors with <i>AvailabilityState</i> <i>Available</i> . | 6. The Test System responds accordingly. |
| 7. Execute Reusable State <i>ParkingBayUnoccupied</i> for <Configured connectorId> <u>Note(s):</u> - The Charging Station will proceed to this end state. This will cause the transaction to stop. - The Charging Station will start the firmware update process the moment the transaction ends or when all interactions with the EV Driver are done (So after the cable has been unplugged, if there is no parking bay sensor). | |

| Main (Test scenario) | |
|---|---|
| 8. The Charging Station sends a FirmwareStatusNotificationRequest With status <i>Downloading</i> | 9. The Test System responds with a FirmwareStatusNotificationResponse |
| 10. The Charging Station sends a FirmwareStatusNotificationRequest With status <i>Downloaded</i> | 11. The Test System responds with a FirmwareStatusNotificationResponse |
| 12. The Charging Station sends a FirmwareStatusNotificationRequest With status <i>SignatureVerified</i> | 13. The Test System responds with a FirmwareStatusNotificationResponse |
| 14. The Charging Station notifies the CSMS about the current state of all connectors. <u>Note(s):</u> - This step is optional. The Charging Station may wants to set its last connector also to <i>Unavailable</i> , before proceeding installing the new firmware. | 15. The Test System responds accordingly. |
| 16. Execute Reusable State <i>RebootBeforeFirmwareInstallation</i> <u>Note:</u> This step only needs to be executed if the Charging Station needs to reboot before firmware <u>installation</u> . | |
| 17. The Charging Station sends a FirmwareStatusNotificationRequest With status <i>Installing</i> <u>Note(s):</u> - This step only needs to be executed if the Charging Station did NOT reboot before firmware <u>installation</u> , at step 16. | 18. The Test System responds with a FirmwareStatusNotificationResponse |
| 19. Execute Reusable State <i>RebootBeforeFirmwareActivation</i> <u>Note:</u> This step only needs to be executed if the Charging Station needs to reboot before firmware <u>activation</u> . | |
| 20. The Test System waits for the Charging Station to reconnect. <u>Note:</u> This step only needs to be executed if the Charging Station did not reboot/reconnect up until this point. The Charging Station should at least reconnect to reestablish the protocol version handshake. <u>Note:</u> Step 21 through 26 can be send in a different order. | |
| 21. The Charging Station notifies the CSMS about the current state of all connectors. <u>Note(s):</u> - This step only needs to be executed if the connectors were previously set to <i>Unavailable</i> (at step 14) and the Charging Station did not report setting them back to <i>Available</i> (after a reboot sequence at step 16 or 19) yet. | 22. The Test System responds accordingly. |
| 23. The Charging Station sends a FirmwareStatusNotificationRequest With status <i>Installed</i> | 24. The Test System responds with a FirmwareStatusNotificationResponse |
| 25. The Charging Station sends a SecurityEventNotificationRequest With type <i>FirmwareUpdated</i> | 26. The Test System responds with a SecurityEventNotificationResponse |

| Tool validations |
|---|
| <p>* Step 2:</p> <p>Message UpdateFirmwareResponse</p> <ul style="list-style-type: none"> - status <i>Accepted</i> <p>* Step 3:</p> <p>Message FirmwareStatusNotificationRequest</p> <ul style="list-style-type: none"> - status <i>DownloadScheduled</i> <p>* Step 5:</p> <p>Message: StatusNotificationRequest</p> <ul style="list-style-type: none"> - connectorStatus <i>Unavailable</i> <p>Or</p> <p>Message: NotifyEventRequest</p> <ul style="list-style-type: none"> - eventData[0].trigger <i>Delta</i> - eventData[0].actualValue <i>"Unavailable"</i> - eventData[0].component.name <i>"Connector"</i> - eventData[0].variable.name <i>"AvailabilityState"</i> <p>* Step 8:</p> <p>Message FirmwareStatusNotificationRequest</p> <ul style="list-style-type: none"> - status <i>Downloading</i> <p>* Step 10:</p> <p>Message FirmwareStatusNotificationRequest</p> <ul style="list-style-type: none"> - status <i>Downloaded</i> <p>* Step 12:</p> <p>Message FirmwareStatusNotificationRequest</p> <ul style="list-style-type: none"> - status <i>SignatureVerified</i> <p>* Step 14:</p> <p>Message: StatusNotificationRequest</p> <ul style="list-style-type: none"> - connectorStatus <i>Unavailable</i> <p>Or</p> <p>Message: NotifyEventRequest</p> <ul style="list-style-type: none"> - eventData[0].trigger <i>Delta</i> - eventData[0].actualValue <i>"Unavailable"</i> - eventData[0].component.name <i>"Connector"</i> - eventData[0].variable.name <i>"AvailabilityState"</i> <p>* Step 17:</p> <p>Message FirmwareStatusNotificationRequest</p> <ul style="list-style-type: none"> - status <i>Installing</i> <p>* Step 21:</p> <p>Message: StatusNotificationRequest</p> <ul style="list-style-type: none"> - connectorStatus <i>Available</i> <p>Or</p> <p>Message: NotifyEventRequest</p> <ul style="list-style-type: none"> - eventData[0].trigger <i>Delta</i> - eventData[0].actualValue <i>"Available"</i> - eventData[0].component.name <i>"Connector"</i> - eventData[0].variable.name <i>"AvailabilityState"</i> <p>* Step 23:</p> <p>Message FirmwareStatusNotificationRequest</p> <ul style="list-style-type: none"> - status <i>Installed</i> <p>* Step 25:</p> <p>Message SecurityEventNotificationRequest</p> <ul style="list-style-type: none"> - type <i>FirmwareUpdated</i> |
| <p>Post scenario validations:</p> <p>N/a</p> |

TC_L_14_CS: Secure Firmware Update - Unable to install and activate firmware with ongoing transaction - AllowNewSessionsPendingFirmwareUpdate is true

| | |
|-------------------|--|
| Test case name | Secure Firmware Update - Unable to install and activate firmware with ongoing transaction - AllowNewSessionsPendingFirmwareUpdate is true |
| Test case Id | TC_L_14_CS |
| Use case Id(s) | L01 |
| Requirement(s) | L01.FR.01,L01.FR.06,L01.FR.07,L01.FR.10,L01.FR.20 |
| System under test | Charging Station |
| Description | The CSMS is able to request the Charging Station to securely download and install/activate a new firmware by sending an <code>UpdateFirmwareRequest</code> with a <code>signingCertificate</code> . When the <i>Installing</i> phase is not possible while a transaction is ongoing, Charging Station will report <i>InstallScheduled</i> and wait for transaction(s) to finish first, else it will immediately report <i>Installing</i> . In both cases before activation of new firmware by (optional) reboot and a reconnect, Charging Station will always wait for transaction(s) to finish. |
| Purpose | To verify if the Charging Station is able to keep allowing new transactions when requested to update the firmware, while there is an ongoing transaction. |
| Prerequisite(s) | <ul style="list-style-type: none"> - A file server has been setup according to the (by the Charging Station) supported file transfer protocol(s), indicated by the configuration variable FileTransferProtocols. - The Charging Station is able to start more than one transaction at a time. - The Charging Station is unable to install and/or activate firmware while there is an ongoing transaction. |

| |
|---|
| Before (Preparations) |
| Configuration State: AllowNewSessionsPendingFirmwareUpdate is <i>true</i> (If implemented) |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> for EVSEId 1 and ConnectorId 1 |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a UpdateFirmwareResponse With status <i>Accepted</i> | 1. The Test System sends a UpdateFirmwareRequest with firmware.installDateTime <Current DateTime - 2 hours> firmware.location <Configured firmware_location> firmware.retrieveDateTime <Current DateTime - 2 hours> firmware.signingCertificate <Configured signingCertificate> firmware.signature <Configured signature> |
| 3. The Charging Station sends a FirmwareStatusNotificationRequest With status <i>Downloading</i> | 4. The Test System responds with a FirmwareStatusNotificationResponse |
| 5. The Charging Station sends a FirmwareStatusNotificationRequest With status <i>Downloaded</i> | 6. The Test System responds with a FirmwareStatusNotificationResponse |
| 7. The Charging Station sends a FirmwareStatusNotificationRequest With status <i>SignatureVerified</i> | 8. The Test System responds with a FirmwareStatusNotificationResponse |
| 9. The Charging Station sends a FirmwareStatusNotificationRequest With status <i>InstallScheduled</i> or status <i>Installing</i> | 10. The Test System responds with a FirmwareStatusNotificationResponse |
| Note(s): - <i>InstallScheduled</i> only applies when Charging Station is not able to install while a transaction is active. | |

| | |
|--|--|
| Main (Test scenario) | |
| 11. Execute Reusable State <i>EnergyTransferStarted</i> for <Configured second Connector> | |
| <u>Note(s):</u> - It is allowed to start a second transaction while there is a (scheduled) firmware update. | |
| 11a. If Charging Station reported <i>Installing</i> in step 9 then wait a while (30-60 s) before continuing with next steps to stop transactions to allow time to install firmware. | |
| 12. Execute Reusable State <i>ParkingBayUnoccupied</i> for <Configured connectorId> | |
| <u>Note(s):</u> - The Charging Station will proceed to this end state. This will cause the first transaction to stop. | |
| 13. Execute Reusable State <i>ParkingBayUnoccupied</i> for <Configured second Connector> | |
| <u>Note(s):</u> - The Charging Station will proceed to this end state. This will cause the second transaction to stop. - The Charging Station will start the firmware update process (if it had not started installing in step 9) the moment this second transaction ends or when all interactions with the EV Driver are done (so after the cable has been unplugged, assuming there is no parking bay sensor). | |
| 14. The Charging Station notifies the CSMS about the current state of all connectors. | 15. The Test System responds accordingly. |
| <u>Note(s):</u> - This step is optional. The Charging Station may want to set its connectors to <i>Unavailable</i> , before proceeding installing the new firmware. | |
| 16. Execute Reusable State <i>RebootBeforeFirmwareInstallation</i> | |
| <u>Note:</u> This step only needs to be executed if the Charging Station needs to reboot before firmware <u>installation</u> . | |
| 17. The Charging Station sends a FirmwareStatusNotificationRequest With status <i>Installing</i> | 18. The Test System responds with a FirmwareStatusNotificationResponse |
| <u>Note(s):</u> - This step only needs to be executed if the Charging Station did not report <i>Installing</i> at step 9 and did not reboot before firmware <u>installation</u> , at step 16 (because that step already reports <i>Installing</i>). | |
| 19. Execute Reusable State <i>RebootBeforeFirmwareActivation</i> | |
| <u>Note:</u> This step only needs to be executed if the Charging Station needs to reboot before firmware <u>activation</u> . | |
| 20. The Test System waits for the Charging Station to reconnect. | |
| <u>Note:</u> This step only needs to be executed if the Charging Station did not reboot/reconnect up until this point. The Charging Station should at least reconnect to reestablish the protocol version handshake. | |
| <u>Note:</u> Step 21 through 26 can be sent in a different order. | |

| Main (Test scenario) | |
|--|---|
| <p>21. The Charging Station notifies the CSMS about the current state of all connectors.</p> <p><u>Note(s):</u> - This step only needs to be executed if the connectors were previously set to <i>Unavailable</i> (at step 14) and the Charging Station did not report setting them back to <i>Available</i> (after a reboot sequence at step 16 or 19) yet.</p> | <p>22. The Test System responds accordingly.</p> |
| <p>23. The Charging Station sends a FirmwareStatusNotificationRequest With status <i>Installed</i></p> | <p>24. The Test System responds with a FirmwareStatusNotificationResponse</p> |
| <p>25. The Charging Station sends a SecurityEventNotificationRequest With type <i>FirmwareUpdated</i></p> | <p>26. The Test System responds with a SecurityEventNotificationResponse</p> |

Tool validations

* Step 2:

Message **UpdateFirmwareResponse**

- **status** *Accepted*

* Step 3:

Message **FirmwareStatusNotificationRequest**

- **status** *Downloading*

* Step 5:

Message **FirmwareStatusNotificationRequest**

- **status** *Downloaded*

* Step 7:

Message **FirmwareStatusNotificationRequest**

- **status** *SignatureVerified*

* Step 9:

Message **FirmwareStatusNotificationRequest**

- **status** *InstallScheduled* or *Installing*

* Step 14: (optional)

Message: **StatusNotificationRequest**

- **connectorStatus** *Unavailable*

Or

Message: **NotifyEventRequest**

- **eventData[0].trigger** *Delta*

- **eventData[0].actualValue** *"Unavailable"*

- **eventData[0].component.name** *"Connector"*

- **eventData[0].variable.name** *"AvailabilityState"*

* Step 17: (optional depending on step 9)

Message **FirmwareStatusNotificationRequest**

- **status** *Installing*

* Step 21:

Message: **StatusNotificationRequest**

- **connectorStatus** *Available*

Or

Message: **NotifyEventRequest**

- **eventData[0].trigger** *Delta*

- **eventData[0].actualValue** *"Available"*

- **eventData[0].component.name** *"Connector"*

- **eventData[0].variable.name** *"AvailabilityState"*

* Step 23:

Message **FirmwareStatusNotificationRequest**

- **status** *Installed*

* Step 25:

Message **SecurityEventNotificationRequest**

- **type** *FirmwareUpdated*

Post scenario validations:

N/a

TC_L_15_CS: Secure Firmware Update - Unable to install and activate firmware with ongoing transaction - AllowNewSessionsPendingFirmwareUpdate is false

| | |
|-------------------|---|
| Test case name | Secure Firmware Update - Unable to install and activate firmware with ongoing transaction - AllowNewSessionsPendingFirmwareUpdate is false |
| Test case Id | TC_L_15_CS |
| Use case Id(s) | L01 |
| Requirement(s) | L01.FR.01,L01.FR.06,L01.FR.07,L01.FR.10,L01.FR.20 |
| System under test | Charging Station |
| Description | The CSMS is able to request the Charging Station to securely download and install a new firmware by sending an <code>UpdateFirmwareRequest</code> with a <code>signingCertificate</code> . When the <i>Installing</i> phase is not possible while a transaction is ongoing, Charging Station will report <i>InstallScheduled</i> and wait for transaction(s) to finish first, else it will immediately report <i>Installing</i> . In both cases before activation of new firmware by (optional) reboot and a reconnect, Charging Station will always wait for transaction(s) to finish. |
| Purpose | To verify if the Charging Station is able to set its available connectors to Unavailable when requested to update the firmware, while there is an ongoing transaction. |
| Prerequisite(s) | <ul style="list-style-type: none"> - A file server has been setup according to the (by the Charging Station) supported file transfer protocol(s), indicated by the configuration variable FileTransferProtocols. - The configuration variable AllowNewSessionsPendingFirmwareUpdate is implemented. - The Charging Station is unable to install and/or activate firmware while there is an ongoing transaction. |

| |
|---|
| Before (Preparations) |
| Configuration State: AllowNewSessionsPendingFirmwareUpdate is <i>false</i> |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a UpdateFirmwareResponse With status <i>Accepted</i> | 1. The Test System sends a UpdateFirmwareRequest with firmware.installDateTime <Current DateTime - 2 hours> firmware.location <Configured firmware_location> firmware.retrieveDateTime <Current DateTime - 2 hours> firmware.signingCertificate <Configured signingCertificate> firmware.signature <Configured signature> |
| 3. The Charging Station sends a FirmwareStatusNotificationRequest With status <i>Downloading</i> | 4. The Test System responds with a FirmwareStatusNotificationResponse |
| 5. The Charging Station sends a FirmwareStatusNotificationRequest With status <i>Downloaded</i> | 6. The Test System responds with a FirmwareStatusNotificationResponse |
| 7. The Charging Station sends a FirmwareStatusNotificationRequest With status <i>SignatureVerified</i> | 8. The Test System responds with a FirmwareStatusNotificationResponse |
| 9. The Charging Station sends a FirmwareStatusNotificationRequest With status <i>InstallScheduled</i> or status <i>Installing</i> | 10. The Test System responds with a FirmwareStatusNotificationResponse |
| Note: <i>InstallScheduled</i> only applies when Charging Station is not able to install while a transaction is active. Part 2 specification only describes that this status needs to be send in case the <i>installDateTime</i> is in the future. However, it is also allowed to send this status if the Charging Station schedules the firmware installation, because of an ongoing transaction. | |

| Main (Test scenario) | |
|---|---|
| <p>11. The Charging Station notifies the CSMS that its Available connector(s) have been set to Unavailable.</p> <p><u>Note(s):</u> - This step needs to be executed for all connectors with AvailabilityState Available.</p> | <p>12. The Test System responds accordingly.</p> |
| <p>12a. If Charging Station reported <i>Installing</i> in step 9 then wait a while (30-60 s) before continuing with next steps to stop transaction to allow time to install firmware.</p> | |
| <p>13. Execute Reusable State <i>ParkingBayUnoccupied</i> for <Configured connectorId></p> <p><u>Note(s):</u> - The Charging Station will proceed to this end state. This will cause the transaction to stop. - The Charging Station will start the firmware update process (if it had not started installing in step 9) the moment the transaction ends or when all interactions with the EV Driver are done (so after the cable has been unplugged, assuming there is no parking bay sensor).</p> | |
| <p>14. The Charging Station notifies the CSMS about the current state of all connectors.</p> <p><u>Note(s):</u> - This step is optional. The Charging Station may want to set its last connector to Unavailable, before proceeding installing the new firmware.</p> | <p>15. The Test System responds accordingly.</p> |
| <p>16. Execute Reusable State <i>RebootBeforeFirmwareInstallation</i></p> <p><u>Note:</u> This step only needs to be executed if the Charging Station needs to reboot before firmware <u>installation</u>.</p> | |
| <p>17. The Charging Station sends a FirmwareStatusNotificationRequest With status <i>Installing</i></p> <p><u>Note(s):</u> - This step only needs to be executed if the Charging Station did not report <i>Installing</i> at step 9 and did not reboot before firmware <u>installation</u>, at step 16 (because that step already reports <i>Installing</i>).</p> | <p>18. The Test System responds with a FirmwareStatusNotificationResponse</p> |
| <p>19. Execute Reusable State <i>RebootBeforeFirmwareActivation</i></p> <p><u>Note:</u> This step only needs to be executed if the Charging Station needs to reboot before firmware <u>activation</u>.</p> | |
| <p>20. The Test System waits for the Charging Station to reconnect.</p> <p><u>Note:</u> This step only needs to be executed if the Charging Station did not reboot/reconnect up until this point. The Charging Station should at least reconnect to reestablish the protocol version handshake.</p> <p><u>Note:</u> Step 21 through 26 can be sent in a different order.</p> | |
| <p>21. The Charging Station notifies the CSMS about the current state of all connectors.</p> <p><u>Note(s):</u> - This step only needs to be executed if the connectors were previously set to Unavailable (at step 14) and the Charging Station did not report setting them back to Available (after a reboot sequence at step 16 or 19) yet.</p> | <p>22. The Test System responds accordingly.</p> |

| Main (Test scenario) | |
|---|---|
| 23. The Charging Station sends a FirmwareStatusNotificationRequest With status <i>Installed</i> | 24. The Test System responds with a FirmwareStatusNotificationResponse |
| 25. The Charging Station sends a SecurityEventNotificationRequest With type <i>FirmwareUpdated</i> | 26. The Test System responds with a SecurityEventNotificationResponse |

| Tool validations |
|--|
| <p>* Step 2: Message UpdateFirmwareResponse - status <i>Accepted</i></p> <p>* Step 3: Message FirmwareStatusNotificationRequest - status <i>Downloading</i></p> <p>* Step 5: Message FirmwareStatusNotificationRequest - status <i>Downloaded</i></p> <p>* Step 7: Message FirmwareStatusNotificationRequest - status <i>SignatureVerified</i></p> <p>* Step 9: Message FirmwareStatusNotificationRequest - status <i>InstallScheduled</i> or <i>Installing</i></p> <p>* Step 11: Message: StatusNotificationRequest - connectorStatus <i>Unavailable</i> Or Message: NotifyEventRequest - eventData[0].trigger <i>Delta</i> - eventData[0].actualValue <i>"Unavailable"</i> - eventData[0].component.name <i>"Connector"</i> - eventData[0].variable.name <i>"AvailabilityState"</i></p> <p>* Step 14: (optional) Message: StatusNotificationRequest - connectorStatus <i>Unavailable</i> Or Message: NotifyEventRequest - eventData[0].trigger <i>Delta</i> - eventData[0].actualValue <i>"Unavailable"</i> - eventData[0].component.name <i>"Connector"</i> - eventData[0].variable.name <i>"AvailabilityState"</i></p> <p>* Step 17: (optional depending on step 9) Message FirmwareStatusNotificationRequest - status <i>Installing</i></p> <p>* Step 21: Message: StatusNotificationRequest - connectorStatus <i>Available</i> Or Message: NotifyEventRequest - eventData[0].trigger <i>Delta</i> - eventData[0].actualValue <i>"Available"</i> - eventData[0].component.name <i>"Connector"</i> - eventData[0].variable.name <i>"AvailabilityState"</i></p> <p>* Step 23: Message FirmwareStatusNotificationRequest - status <i>Installed</i></p> <p>* Step 25: Message SecurityEventNotificationRequest - type <i>FirmwareUpdated</i></p> |
| <p>Post scenario validations: N/a</p> |

TC_L_16_CS: Secure Firmware Update - Able to update firmware with ongoing transaction

| | |
|-------------------|--|
| Test case name | Secure Firmware Update - Able to update firmware with ongoing transaction |
| Test case Id | TC_L_16_CS |
| Use case Id(s) | L01 |
| Requirement(s) | L01.FR.01,L01.FR.06,L01.FR.10,L01.FR.20 |
| System under test | Charging Station |
| Description | The CSMS is able to request the Charging Station to securely download and install a new firmware by sending an UpdateFirmwareRequest with a signingCertificate. |
| Purpose | To verify if the Charging Station is able to securely download and install a new firmware, while a transaction is ongoing. |
| Prerequisite(s) | - A file server has been setup according to the (by the Charging Station) supported file transfer protocol(s), indicated by the configuration variable FileTransferProtocols . - The Charging Station is able to update its firmware while a transaction is ongoing. |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a UpdateFirmwareResponse | 1. The Test System sends a UpdateFirmwareRequest with firmware.installDateTime <Current DateTime - 2 hours> firmware.location <Configured firmware_location> firmware.retrieveDateTime <Current DateTime - 2 hours> firmware.signingCertificate <Configured signingCertificate> firmware.signature <Configured signature> |
| 3. The Charging Station sends a FirmwareStatusNotificationRequest . With status <i>Downloading</i> | 4. The Test System responds with a FirmwareStatusNotificationResponse . |
| 5. The Charging Station sends a FirmwareStatusNotificationRequest . With status <i>Downloaded</i> | 6. The Test System responds with a FirmwareStatusNotificationResponse . |
| 7. The Charging Station sends a FirmwareStatusNotificationRequest . With status <i>SignatureVerified</i> | 8. The Test System responds with a FirmwareStatusNotificationResponse . |
| 9. The Charging Station sends a FirmwareStatusNotificationRequest . With status <i>Installing</i> | 10. The Test System responds with a FirmwareStatusNotificationResponse . |
| 11. The Test System waits for the Charging Station to reconnect. | |
| <u>Note:</u> The Charging Station reconnects to reestablish the protocol version handshake. | |
| 12. The Charging Station sends a FirmwareStatusNotificationRequest . With status <i>Installed</i> | 13. The Test System responds with a FirmwareStatusNotificationResponse . |
| 14. The Charging Station sends a SecurityEventNotificationRequest With type <i>FirmwareUpdated</i> | 15. The Test System responds with a SecurityEventNotificationResponse |

| Tool validations |
|--|
| <div><div>* Step 2:</div><div>Message UpdateFirmwareResponse</div><div>- status <i>Accepted</i></div><div>* Step 3:</div><div>Message FirmwareStatusNotificationRequest</div><div>- status <i>Downloading</i></div><div>* Step 5:</div><div>Message FirmwareStatusNotificationRequest</div><div>- status <i>Downloaded</i></div><div>* Step 7:</div><div>Message FirmwareStatusNotificationRequest</div><div>- status <i>SignatureVerified</i></div><div>* Step 9:</div><div>Message FirmwareStatusNotificationRequest</div><div>- status <i>Installing</i></div><div>* Step 12:</div><div>Message FirmwareStatusNotificationRequest</div><div>- status <i>Installed</i></div><div>* Step 14:</div><div>Message SecurityEventNotificationRequest</div><div>- type <i>FirmwareUpdated</i></div></div> |
| <div><div>Post scenario validations:</div><div>N/a</div></div> |

TC_L_18_CS: Secure Firmware Update - Missing firmware signing certificate and signature

| | |
|-------------------|---|
| Test case name | Secure Firmware Update - Missing firmware signing certificate and signature |
| Test case Id | TC_L_18_CS |
| Use case Id(s) | L01 |
| Requirement(s) | N/a |
| System under test | Charging Station |
| Description | The CSMS is able to request the Charging Station to securely download and install a new firmware by sending an UpdateFirmwareRequest with a signingCertificate. |
| Purpose | To verify if the Charging Station is not accepting a non-secure firmware update request, when supporting secure firmware update. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a UpdateFirmwareResponse | 1. The Test System sends a UpdateFirmwareRequest with firmware.installDateTime <Current DateTime - 2 hours> firmware.location <Configured firmware_location> firmware.retrieveDateTime <Current DateTime - 2 hours> firmware.signingCertificate is omitted firmware.signature is omitted |

| |
|--|
| Tool validations |
| * Step 2: Message UpdateFirmwareResponse - status <i>Rejected</i> OR <i>InvalidCertificate</i> |
| Post scenario validations: N/a |

M CertificateManagement

TC_M_01_CS: Install CA certificate - CSMSRootCertificate

| | |
|-------------------|---|
| Test case name | Install CA certificate - CSMSRootCertificate |
| Test case Id | TC_M_01_CS |
| Use case Id(s) | M05 |
| Requirement(s) | M05.FR.01,M05.FR.02 |
| System under test | Charging Station |
| Description | The CSMS is able to request the Charging Station to install new Root CA certificates using the InstallCertificateRequest message. |
| Purpose | To verify if the Charging Station is able to install a new CSMSRootCertificate. |
| Prerequisite(s) | - The Charging Station supports Security Profile 2 or 3. |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>CertificateInstalled</i> for certificateType CSMSRootCertificate (Root 2) | |
| <u>Note(s):</u> - When the Charging Station has the following configuration; AdditionalRootCertificateCheck implemented with value true , then a custom CSMSRootCertificate should be used. | |
| 2. Execute Reusable State <i>GetInstalledCertificates</i> for certificateType CSMSRootCertificate | |

| |
|-----------------------------------|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_M_02_CS: Install CA certificate - ManufacturerRootCertificate

| | |
|-------------------|---|
| Test case name | Install CA certificate - ManufacturerRootCertificate |
| Test case Id | TC_M_02_CS |
| Use case Id(s) | M05 |
| Requirement(s) | M05.FR.01,M05.FR.02 |
| System under test | Charging Station |
| Description | The CSMS is able to request the Charging Station to install new Root CA certificates using the InstallCertificateRequest message. |
| Purpose | To verify if the Charging Station is able to install a new ManufacturerRootCertificate. |
| Prerequisite(s) | The Charging Station supports signed firmware updates. |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>CertificateInstalled</i> for certificateType <i>ManufacturerRootCertificate</i> | |
| 2. Execute Reusable State <i>GetInstalledCertificates</i> for certificateType <i>ManufacturerRootCertificate</i> | |

| |
|-----------------------------------|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_M_03_CS: Install CA certificate - V2GRootCertificate

| | |
|-------------------|---|
| Test case name | Install CA certificate - V2GRootCertificate |
| Test case Id | TC_M_03_CS |
| Use case Id(s) | M05 |
| Requirement(s) | M05.FR.01,M05.FR.02 |
| System under test | Charging Station |
| Description | The CSMS is able to request the Charging Station to install new Root CA certificates using the InstallCertificateRequest message. |
| Purpose | To verify if the Charging Station is able to install a new V2GRootCertificate. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The Charging Station supports ISO 15118. - The Charging Station does NOT have the following configuration; AdditionalRootCertificateCheck is implemented with value <i>true</i> |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>CertificateInstalled</i> for certificateType V2GRootCertificate | |
| 2. Execute Reusable State <i>GetInstalledCertificates</i> for certificateType V2GRootCertificate | |

| |
|--|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_M_04_CS: Install CA certificate - MORootCertificate

| | |
|-------------------|---|
| Test case name | Install CA certificate - MORootCertificate |
| Test case Id | TC_M_04_CS |
| Use case Id(s) | M05 |
| Requirement(s) | M05.FR.01,M05.FR.02 |
| System under test | Charging Station |
| Description | The CSMS is able to request the Charging Station to install new Root CA certificates using the InstallCertificateRequest message. |
| Purpose | To verify if the Charging Station is able to install a new MORootCertificate. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The Charging Station supports ISO 15118. - The Charging Station does NOT have the following configuration; AdditionalRootCertificateCheck is implemented with value <i>true</i> |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>CertificateInstalled</i> for certificateType <i>MORootCertificate</i> | |
| 2. Execute Reusable State <i>GetInstalledCertificates</i> for certificateType <i>MORootCertificate</i> | |

| |
|--|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_M_07_CS: Install CA certificate - Rejected - Certificate invalid

| | |
|-------------------|---|
| Test case name | Install CA certificate - Rejected - Certificate invalid |
| Test case Id | TC_M_07_CS |
| Use case Id(s) | M05 |
| Requirement(s) | M05.FR.01,M05.FR.07 |
| System under test | Charging Station |
| Description | The CSMS is able to request the Charging Station to install new Root CA certificates using the <code>InstallCertificateRequest</code> message. |
| Purpose | To verify if the Charging Station is able to reject an invalid certificate. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The Charging Station supports Security Profile 2 or 3. - The Charging Station does NOT have the following configuration; AdditionalRootCertificateCheck is implemented with value <i>true</i> |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|---|---|
| Charging Station | CSMS |
| 2. The Charging Station responds with a InstallCertificateResponse | 1. The Test System sends a InstallCertificateRequest with certificateType is <i>CSMSRootCertificate</i> certificate is <i><Generated Expired Certificate></i> |
| 4. The Charging Station responds with a GetInstalledCertificateIdsResponse | 3. The Test System sends a GetInstalledCertificateIdsRequest with certificateType is <i>CSMSRootCertificate</i> |

| |
|--|
| Tool validations |
| <p>* Step 2:</p> <p>Message: InstallCertificateResponse</p> <ul style="list-style-type: none"> - status must be <i>Rejected</i> <p>* Step 4:</p> <p>Message: GetInstalledCertificateIdsResponse</p> <ul style="list-style-type: none"> - status must be <i>NotFound</i> <p>OR</p> <ul style="list-style-type: none"> - status must be <i>Accepted</i> - certificateHashDataChain must NOT contain an entry with following values: <ul style="list-style-type: none"> – certificateType is <i>CSMSRootCertificate</i> – certificateHashData contains <i><HashData from configured new CSMS Root certificate></i> |
| Post scenario validations: N/a |

TC_M_09_CS: Install CA certificate - AdditionalRootCertificateCheck - Rejected

| | |
|-------------------|--|
| Test case name | Install CA certificate - AdditionalRootCertificateCheck - Rejected |
| Test case Id | TC_M_09_CS |
| Use case Id(s) | M05 |
| Requirement(s) | M05.FR.10,M05.FR.11 |
| System under test | Charging Station |
| Description | The CSMS is able to request the Charging Station to install new Root CA certificates using the InstallCertificateRequest message. |
| Purpose | To verify if the Charging Station is able to reject installing a new CSMSRootCertificate that is not signed by the old CSMSRootCertificate, while additional security measures for installing a root certificate is active. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The Charging Station supports Security Profile 2 or 3. - The Charging Station has the configuration variable AdditionalRootCertificateCheck implemented with value <i>true</i> |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a InstallCertificateResponse | 1. The Test System sends a InstallCertificateRequest with certificateType is <i>CSMSRootCertificate</i> certificate is <i><Configured CSMSRootCertificate></i> <u>Note(s):</u> - <i>CSMSRootCertificate must have not been signed by old certificate.</i> |
| 4. The Charging Station responds with a GetInstalledCertificateIdsResponse | 3. The Test System sends a GetInstalledCertificateIdsRequest with certificateType is <i>CSMSRootCertificate</i> |

| |
|--|
| Tool validations |
| * Step 2: Message: InstallCertificateResponse - status must be <i>Rejected</i> * Step 4: Message: GetInstalledCertificateIdsResponse - status must be <i>Accepted</i> - certificateHashDataChain must contain one entry with following values: - certificateType is <i>CSMSRootCertificate</i> - certificateHashData contains <i><HashData from configured old CSMS Root certificate></i> |
| Post scenario validations: N/a |

TC_M_30_CS: Install CA certificate - AdditionalRootCertificateCheck - Reconnect using new CSMS Root - Success

| | |
|-------------------|--|
| Test case name | Install CA certificate - AdditionalRootCertificateCheck - Reconnect using new CSMS Root - Success |
| Test case Id | TC_M_30_CS |
| Use case Id(s) | M05 |
| Requirement(s) | M05.FR.13 |
| System under test | Charging Station |
| Description | The CSMS is able to request the Charging Station to install new Root CA certificates using the InstallCertificateRequest message. |
| Purpose | To verify if the Charging Station is able to reconnect to the CSMS, while using a new CSMS Root certificate. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The Charging Station supports Security Profile 2 or 3. - The Charging Station has the configuration variable AdditionalRootCertificateCheck implemented with value <i>true</i> - The at the Test System configured new CSMSRootCertificate must be signed by the old CSMS Root certificate. |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: <i>CertificateInstalled</i> for certificateType <i>CSMSRootCertificate</i> and certificate <Configured new CSMS Root certificate 2> If security profile 3 is enabled, then: <i>RenewChargingStationCertificate</i> for certificateType <i>ChargingStationCertificate</i> |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a ResetResponse | 1. The Test System sends a ResetRequest with type OnIdle |
| 4. During the TLS handshake the Charging Station validates the CSMS certificate. | 3. During the TLS handshake the Test System provides a CSMS certificate which is signed by the <Configured new CSMS Root certificate> |
| Note(s): - This connection attempt must succeed. | |
| 5. Execute Reusable State <i>Booted</i> | |
| 7. The Charging Station responds with a GetInstalledCertificateIdsResponse | 6. The Test System sends a GetInstalledCertificateIdsRequest with certificateType is <i>CSMSRootCertificate</i> |

| |
|--|
| Tool validations |
| * Step 2: Message ResetResponse - status <i>Accepted</i> * Step 7: Message: GetInstalledCertificateIdsResponse - status must be <i>Accepted</i> - certificateHashDataChain must NOT contain an entry with following values: - certificateType is <i>CSMSRootCertificate</i> - certificateHashData contains <HashData from configured old CSMS Root certificate> |
| Post scenario validations: - N/a |

TC_M_31_CS: Install CA certificate - AdditionalRootCertificateCheck - Reconnect using new CSMS Root - Fallback mechanism

| | |
|-------------------|---|
| Test case name | Install CA certificate - AdditionalRootCertificateCheck - Reconnect using new CSMS Root - Fallback mechanism |
| Test case Id | TC_M_31_CS |
| Use case Id(s) | M05 |
| Requirement(s) | M05.FR.14 |
| System under test | Charging Station |
| Description | The CSMS is able to request the Charging Station to install new Root CA certificates using the InstallCertificateRequest message. |
| Purpose | To verify if the Charging Station is able to reconnect to the CSMS using the old CSMS Root certificate, when validating the CSMS certificate using the new CSMS Root certificate fails. |
| Prerequisite(s) | - The Charging Station supports Security Profile 2 or 3. - The Charging Station has the configuration variable AdditionalRootCertificateCheck implemented with value <i>true</i> - The at the Test System configured new CSMSRootCertificate must be signed by the old CSMS Root certificate. |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: <i>CertificateInstalled</i> for certificateType <i>CSMSRootCertificate</i> and certificate <Configured (new) CSMS Root certificate 2> |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a ResetResponse | 1. The Test System sends a ResetRequest with type OnIdle |
| 4. During the TLS handshake the Charging Station validates the CSMS certificate. <u>Note(s):</u> - This connection attempt fails, because the Charging Station will use the new CSMS Root certificate to validate the CSMS certificate. | 3. During the TLS handshake the Test System provides a CSMS certificate which is signed by the <Configured old CSMS Root certificate> |
| 5. The Charging Station re-validates the CSMS certificate. <u>Note(s):</u> - This connection attempt succeeds, because the Charging Station will now use the old CSMS Root certificate to validate the CSMS certificate. | |
| 6. Execute Reusable State <i>Booted</i> | |
| 8. The Charging Station responds with a GetInstalledCertificateIdsResponse | 7. The Test System sends a GetInstalledCertificateIdsRequest with certificateType is <i>CSMSRootCertificate</i> |

| Tool validations |
|---|
| <div>* Step 2: Message ResetResponse - status <i>Accepted</i></div> <div>* Step 8: Message: GetInstalledCertificateIdsResponse - status must be <i>Accepted</i> - certificateHashDataChain must contain an entry with following values: - certificateType is <i>CSMSRootCertificate</i> - certificateHashData contains <i><HashData from configured old CSMS Root certificate></i></div> |
| <div>Post scenario validations: - N/a</div> |

TC_M_12_CS: Retrieve certificates from Charging Station - CSMSRootCertificate

| | |
|--------------------------|--|
| Test case name | Retrieve certificates from Charging Station - CSMSRootCertificate |
| Test case Id | TC_M_12_CS |
| Use case Id(s) | M03 |
| Requirement(s) | M03.FR.01,M03.FR.03,M03.FR.04 |
| System under test | Charging Station |
| Description | The CSMS is able to retrieve the certificates installed at the Charging Station using the GetInstalledCertificateIdsRequest message. |
| Purpose | To verify if the Charging Station is able to provide the hashData from all stored CSMSRootCertificates. |
| Prerequisite(s) | - The Charging Station supports Security Profile 2 or 3. |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: <i>GetInstalledCertificates</i> from certificateType CSMSRootCertificate |
| Reusable State(s): N/a |

| | |
|---|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>GetInstalledCertificates</i> for certificateType CSMSRootCertificate | |

| |
|--|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_M_13_CS: Retrieve certificates from Charging Station - ManufacturerRootCertificate

| | |
|-------------------|--|
| Test case name | Retrieve certificates from Charging Station - ManufacturerRootCertificate |
| Test case Id | TC_M_13_CS |
| Use case Id(s) | M03 |
| Requirement(s) | M03.FR.01,M03.FR.03,M03.FR.04 |
| System under test | Charging Station |
| Description | The CSMS is able to retrieve the certificates installed at the Charging Station using the GetInstalledCertificateIdsRequest message. |
| Purpose | To verify if the Charging Station is able to provide the hashData from all stored ManufacturerRootCertificate. |
| Prerequisite(s) | - The Charging Station supports signed firmware updates. |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: CertificateInstalled from certificateType <i>ManufacturerRootCertificate</i> |
| Reusable State(s): N/a |

| | |
|---|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>GetInstalledCertificates</i> for certificateType <i>ManufacturerRootCertificate</i> | |

| |
|-----------------------------------|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_M_14_CS: Retrieve certificates from Charging Station - V2GRootCertificate

| | |
|-------------------|--|
| Test case name | Retrieve certificates from Charging Station - V2GRootCertificate |
| Test case Id | TC_M_14_CS |
| Use case Id(s) | M03 |
| Requirement(s) | M03.FR.01,M03.FR.03,M03.FR.04 |
| System under test | Charging Station |
| Description | The CSMS is able to retrieve the certificates installed at the Charging Station using the GetInstalledCertificateIdsRequest message. |
| Purpose | To verify if the Charging Station is able to provide the hashData from all stored V2GRootCertificate. |
| Prerequisite(s) | The Charging Station supports ISO 15118. |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: CertificateInstalled from certificateType V2GRootCertificate |
| Reusable State(s): N/a |

| | |
|---|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>GetInstalledCertificates</i> for certificateType V2GRootCertificate | |

| |
|--|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_M_15_CS: Retrieve certificates from Charging Station - V2GCertificateChain

| | |
|-------------------|---|
| Test case name | Retrieve certificates from Charging Station - V2GCertificateChain |
| Test case Id | TC_M_15_CS |
| Use case Id(s) | M03 |
| Requirement(s) | M03.FR.01,M03.FR.03,M03.FR.04,M03.FR.05 |
| System under test | Charging Station |
| Description | The CSMS is able to retrieve the certificates installed at the Charging Station using the GetInstalledCertificateIdsRequest message. |
| Purpose | To verify if the Charging Station is able to provide the hashData from all stored certificates that are part of a V2GCertificateChain. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The Charging Station supports ISO 15118. - The Charging Station has atleast one V2GCertificateChain installed. |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: RenewV2GChargingStationCertificate |
| Reusable State(s): N/a |

| | |
|---|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State GetInstalledCertificates for certificateType V2GCertificateChain | |

| |
|---|
| Tool validations |
| <p>* Step 1:</p> <p>Message: GetInstalledCertificateIdsResponse</p> <ul style="list-style-type: none"> - status must be <i>Accepted</i> - certificateHashDataChain must contain an entry with following values: <p><i>Note: Order does not matter.</i></p> <ul style="list-style-type: none"> - certificateType is V2GCertificateChain - certificateHashData uses the childCertificateHashData field |
| Post scenario validations: N/a |

TC_M_16_CS: Retrieve certificates from Charging Station - MORootCertificate

| | |
|-------------------|--|
| Test case name | Retrieve certificates from Charging Station - MORootCertificate |
| Test case Id | TC_M_16_CS |
| Use case Id(s) | M03 |
| Requirement(s) | M03.FR.01,M03.FR.03,M03.FR.04 |
| System under test | Charging Station |
| Description | The CSMS is able to retrieve the certificates installed at the Charging Station using the GetInstalledCertificateIdsRequest message. |
| Purpose | To verify if the Charging Station is able to provide the hashData from all stored MORootCertificate. |
| Prerequisite(s) | The Charging Station supports ISO 15118. |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: <i>CertificateInstalled</i> from certificateType <i>MORootCertificate</i> |
| Reusable State(s): N/a |

| | |
|---|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>GetInstalledCertificates</i> for certificateType <i>MORootCertificate</i> | |

| |
|-----------------------------------|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_M_17_CS: Retrieve certificates from Charging Station - CSMSRootCertificate & ManufacturerRootCertificate

| | |
|-------------------|---|
| Test case name | Retrieve certificates from Charging Station - CSMSRootCertificate & ManufacturerRootCertificate |
| Test case Id | TC_M_17_CS |
| Use case Id(s) | M03 |
| Requirement(s) | M03.FR.01,M03.FR.03,M03.FR.04 |
| System under test | Charging Station |
| Description | The CSMS is able to retrieve the certificates installed at the Charging Station using the GetInstalledCertificateIdsRequest message. |
| Purpose | To verify if the Charging Station is able to provide the hashData from all stored CSMSRootCertificates and ManufacturerRootCertificates |
| Prerequisite(s) | <ul style="list-style-type: none"> - The Charging Station supports Security Profile 2 or 3. - The Charging Station supports signed firmware updates. |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: <i>CertificateInstalled</i> from certificateType CSMSRootCertificate <i>CertificateInstalled</i> from certificateType ManufacturerRootCertificate |
| Reusable State(s): N/a |

| | |
|--|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>GetInstalledCertificates</i> for certificateType CSMSRootCertificate AND ManufacturerRootCertificate | |

| |
|--|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_M_18_CS: Retrieve certificates from Charging Station - All certificateTypes

| | |
|-------------------|---|
| Test case name | Retrieve certificates from Charging Station - All certificateTypes |
| Test case Id | TC_M_18_CS |
| Use case Id(s) | M03 |
| Requirement(s) | M03.FR.01,M03.FR.03,M03.FR.04 |
| System under test | Charging Station |
| Description | The CSMS is able to retrieve the certificates installed at the Charging Station using the GetInstalledCertificateIdsRequest message. |
| Purpose | To verify if the Charging Station is able to provide the hashData from all stored certificates |
| Prerequisite(s) | <ul style="list-style-type: none"> - The Charging Station supports Security Profile 2 or 3. - The Charging Station supports signed firmware updates. |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: <i>CertificateInstalled</i> from certificateType CSMSRootCertificate <i>CertificateInstalled</i> from certificateType ManufacturerRootCertificate |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|---|--|
| Charging Station | CSMS |
| 2. The Charging Station responds with a GetInstalledCertificateIdsResponse | 1. The Test System sends a GetInstalledCertificateIdsRequest With certificateType is omitted. |

| |
|--|
| Tool validations |
| <p>* Step 2:</p> <p>Message: GetInstalledCertificateIdsResponse</p> <ul style="list-style-type: none"> - status must be <i>Accepted</i> - certificateHashDataChain must contain the following two entries with following values: <p>Note: Order does not matter.</p> <p>Entry 1:</p> <ul style="list-style-type: none"> - certificateHashDataChain[0].certificateType is <i>CSMSRootCertificate</i> - certificateHashDataChain[0].certificateHashData contains <i><HashData from configured new CSMS Root certificate></i> <p>Entry 2:</p> <ul style="list-style-type: none"> - certificateHashDataChain[1].certificateType is <i>ManufacturerRootCertificate</i> - certificateHashDataChain[1].certificateHashData contains <i><HashData from configured new Manufacturer Root certificate></i> <p>Post scenario validations: N/a </p> |

TC_M_19_CS: Retrieve certificates from Charging Station - No matching certificate found

| | |
|--------------------------|---|
| Test case name | Retrieve certificates from Charging Station - No matching certificate found |
| Test case Id | TC_M_19_CS |
| Use case Id(s) | M03 |
| Requirement(s) | M03.FR.01,M03.FR.02 |
| System under test | Charging Station |
| Description | The CSMS is able to retrieve the certificates installed at the Charging Station using the <code>GetInstalledCertificateIdsRequest</code> message. |
| Purpose | To verify if the Charging Station is able to respond that it did not find any certificate of the requested <code>certificateType</code> . |
| Prerequisite(s) | The Charging Station does not have a <code>MORootCertificate</code> installed, or it must be possible to remove it. |

| |
|---|
| Before (Preparations) |
| Configuration State: Test System checks to make sure that no <code>MORootCertificate</code> is installed via <code>GetInstalledCertificateIds</code> . If an <code>MORootCertificate</code> exists it removes it via <code>DeleteCertificate</code> . |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a <code>GetInstalledCertificateIdsResponse</code> | 1. The Test System sends a <code>GetInstalledCertificateIdsRequest</code> With <code>certificateType</code> is <i>MORootCertificate</i> |

| |
|---|
| Tool validations |
| * Step 2: Message: <code>GetInstalledCertificateIdsResponse</code> - <code>status</code> must be <i>NotFound</i> - <code>certificateHashDataChain</code> must be omitted. |
| Post scenario validations: N/a |

TC_M_20_CS: Delete a certificate from a Charging Station - Success

| | |
|-------------------|---|
| Test case name | Delete a certificate from a Charging Station - Success |
| Test case Id | TC_M_20_CS |
| Use case Id(s) | M04 |
| Requirement(s) | M04.FR.01,M04.FR.02 |
| System under test | Charging Station |
| Description | The CSMS is able to request the Charging Station to delete an installed certificate using the DeleteCertificateRequest message. |
| Purpose | To verify if the Charging Station is able to delete an installed certificate. |
| Prerequisite(s) | - The Charging Station supports Security Profile 2 or 3. |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): GetInstalledCertificates with certificateType CSMSRootCertificate CertificateInstalled with certificateType CSMSRootCertificate (When no certificate is returned at GetInstalledCertificates) |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>GetInstalledCertificates</i> with certificateType CSMSRootCertificate | |
| 3. The Charging Station responds with a DeleteCertificateResponse | 2. The Test System sends a DeleteCertificateRequest with certificateHashData contains <i><Returned certificateHashData at step 1></i> |
| 4. Execute Reusable State <i>GetInstalledCertificates</i> with certificateType CSMSRootCertificate | |

| |
|--|
| Tool validations |
| <p>* Step 1:</p> <ul style="list-style-type: none"> - Certificate that is going to be deleted is present. <p>* Step 3:</p> <p>Message: DeleteCertificateResponse</p> <ul style="list-style-type: none"> - status must be <i>Accepted</i> <p>* Step 4:</p> <ul style="list-style-type: none"> - Certificate that should be deleted is not present anymore. |
| Post scenario validations: N/a |

TC_M_22_CS: Delete a certificate from a Charging Station - No matching certificate found

| | |
|-------------------|--|
| Test case name | Delete a certificate from a Charging Station - No matching certificate found |
| Test case Id | TC_M_22_CS |
| Use case Id(s) | M04 |
| Requirement(s) | M04.FR.01,M04.FR.04 |
| System under test | Charging Station |
| Description | The CSMS is able to request the Charging Station to delete an installed certificate using the DeleteCertificateRequest message. |
| Purpose | To verify if the Charging Station is able to respond that no certificate is installed that matches the provided certificateHashData. |
| Prerequisite(s) | - The Charging Station supports Security Profile 2 or 3. |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>GetInstalledCertificates</i> with certificateType <i>CSMSRootCertificate</i> . | |
| 3. The Charging Station responds with a DeleteCertificateResponse | 2. The Test System sends a DeleteCertificateRequest with certificateHashData is <i><certificateHashData from unknown certificate></i> |

| |
|---|
| Tool validations |
| * Step 3: Message: DeleteCertificateResponse - status must be <i>NotFound</i> |
| Post scenario validations: N/a |

TC_M_23_CS: Delete a certificate from a Charging Station - Unable to delete the Charging Station Certificate

| | |
|-------------------|---|
| Test case name | Delete a certificate from a Charging Station - Unable to delete the Charging Station Certificate |
| Test case Id | TC_M_23_CS |
| Use case Id(s) | M04 |
| Requirement(s) | M04.FR.01,M04.FR.06 |
| System under test | Charging Station |
| Description | The CSMS is able to request the Charging Station to delete an installed certificate using the DeleteCertificateRequest message. |
| Purpose | To verify if the Charging Station does NOT allow the deletion of the Charging Station certificate. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The Charging Station supports Security Profile 3. - A valid <i>CSMSRootCertificate</i> is installed on the Charging Station. |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): RenewChargingStationCertificate for certificateType <i>ChargingStationCertificate</i> |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>GetInstalledCertificates</i> with certificateType <i>omitted</i> . | |
| 3. The Charging Station responds with a DeleteCertificateResponse | 2. The Test System sends a DeleteCertificateRequest with certificateHashData is <i><certificateHashData from the generated ChargingStationCertificate at before.></i> |

| |
|--|
| Tool validations |
| * Step 3: Message: DeleteCertificateResponse - status must be <i>NotFound</i> OR <i>Failed</i> |
| Post scenario validations: N/a |

TC_M_24_CS: Get Charging Station Certificate status - Success

| | |
|-------------------|---|
| Test case name | Get Charging Station Certificate status - Success |
| Test case Id | TC_M_24_CS |
| Use case Id(s) | M06 |
| Requirement(s) | M06.FR.06,M06.FR.07 |
| System under test | Charging Station |
| Description | The Charging Station is able to request the CSMS to get the status of a (V2G) Charging Station certificate. |
| Purpose | To verify if the Charging Station is able to request the status of a (V2G) Charging Station certificate. |
| Prerequisite(s) | - The Charging Station supports ISO 15118. |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: CertificateInstalled from certificateType V2GRootCertificate CertificateInstalled from certificateType MORootCertificate RenewV2GChargingStationCertificate |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Charging Station sends a GetCertificateStatusRequest | 2. The Test System responds with a GetCertificateStatusResponse with status <i>Accepted</i> ocspResult <i><OCSPResponse class as defined in IETF RFC 6960. DER encoded (as defined in IETF RFC 6960), and then base64 encoded.></i> |
| <u>Note:</u> Step 1/2 are repeated for the V2G Charging Station (leaf), the subCA1 and subCA2 certificates. | |

| |
|-----------------------------------|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_M_25_CS: Get Charging Station Certificate status - Rejected

| | |
|-------------------|--|
| Test case name | Get Charging Station Certificate status - Rejected |
| Test case Id | TC_M_25_CS |
| Use case Id(s) | M06 |
| Requirement(s) | M06.FR.04 |
| System under test | Charging Station |
| Description | The Charging Station is able to request the CSMS to get the status of a (V2G) Charging Station certificate. |
| Purpose | To verify if the Charging Station is able to handle receiving a rejected status after requesting the status of a (V2G) Charging Station certificate. |
| Prerequisite(s) | - The Charging Station supports ISO 15118. |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: CertificateInstalled from certificateType <i>V2GRootCertificate</i> CertificateInstalled from certificateType <i>MORootCertificate</i> RenewV2GChargingStationCertificate |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|--|--|
| Charging Station | CSMS |
| 1. The Charging Station sends a GetCertificateStatusRequest | 2. The Test System responds with a GetCertificateStatusResponse with status <i>Failed</i> ocspResult is omitted. |

| |
|--|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_M_26_CS: Certificate Installation EV - ISO15118-2 - Success

| | |
|-------------------|---|
| Test case name | Certificate Installation EV - Success |
| Test case Id | TC_M_26_CS |
| Use case Id(s) | M01 |
| Requirement(s) | M01.FR.01 |
| System under test | Charging Station |
| Description | The EV initiates installing a new certificate. The Charging Station forwards the request for a new certificate to the CSMS. |
| Purpose | To verify if the Charging Station is able to forward the request to the CSMS. |
| Prerequisite(s) | - The Charging Station supports ISO 15118-2. |

Before (Preparations)

Configuration State:

-The test case calls *SendISO15118AuthorizationMethod* method with parameter *PnC* in order to inform the EV emulator about the expected authorization method.
 -The test case calls *SendInstallISO15118CertificateMethod* method in order to trigger the EV emulator to initiate installing a new certificate.

Memory State:

CertificateInstalled from certificateType *V2GRootCertificate*
CertificateInstalled from certificateType *MORootCertificate*

Reusable State(s):

State is *EVConnectedPreSession*

Main (Test scenario)

| Charging Station | CSMS |
|---|---|
| 1. The Charging Station sends a Get15118EVCertificateRequest | 2. The Test System responds with a Get15118EVCertificateResponse with status Accepted exiResponse <Raw CertificateInstallationRes response for the EV, Base64 encoded.> |
| 3. The Charging Station sends an AuthorizeRequest | 4. The Test System responds with an AuthorizeResponse with status Accepted |

Tool validations

* Step 1:

Message: **Get15118EVCertificateRequest**

- **action** must be *Install*

Post scenario validations:

N/a

TC_M_27_CS: Certificate Installation EV - Failed

| | |
|-------------------|---|
| Test case name | Certificate Installation EV - Failed |
| Test case Id | TC_M_27_CS |
| Use case Id(s) | M01 |
| Requirement(s) | M01.FR.01, M01.FR.02 |
| System under test | Charging Station |
| Description | The EV initiates installing a new certificate. The Charging Station forwards the request for a new certificate to the CSMS. |
| Purpose | To verify if the Charging Station is able to handle receiving a Failed status. |
| Prerequisite(s) | The Charging Station supports ISO 15118-2. |

| | |
|--|--|
| Before (Preparations) | |
| Configuration State: -The test case calls <i>SendISO15118AuthorizationMethod</i> method with parameter <i>PnC</i> in order to inform the EV emulator about the expected authorization method. -The test case calls <i>SendInstallISO15118CertificateMethod</i> method in order to trigger the EV emulator to initiate installing a new certificate. | |
| Memory State: <i>CertificateInstalled</i> from certificateType <i>V2GRootCertificate</i> <i>CertificateInstalled</i> from certificateType <i>MORootCertificate</i> | |
| Reusable State(s): State is <i>EVConnectedPreSession</i> | |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Charging Station sends a Get15118EVCertificateRequest | 2. The Test System responds with a Get15118EVCertificateResponse with status Failed exiResponse is omitted |
| | 3. If an AuthorizeRequest is received, the testcase will FAIL and the Test System reports why it failed. |

| |
|---|
| Tool validations |
| * Step 1: Message: Get15118EVCertificateRequest - action must be <i>Install</i> |
| Post scenario validations: N/a |

TC_M_28_CS: Certificate Update EV - Success

| | |
|-------------------|---|
| Test case name | Certificate Update EV - Success |
| Test case Id | TC_M_28_CS |
| Use case Id(s) | M02 |
| Requirement(s) | M02.FR.01 |
| System under test | Charging Station |
| Description | The EV initiates updating the existing certificate. The Charging Station forwards the update request to the CSMS. |
| Purpose | To verify if the Charging Station is able to forward the request to the CSMS. |
| Prerequisite(s) | The Charging Station supports ISO 15118-2. |

| |
|---|
| Before (Preparations) |
| <p>Configuration State: ISO15118Ctrlr.ContractCertificateInstallationEnabled is <i>true</i> -The test case calls <i>SendISO15118AuthorizationMethod</i> method with parameter <i>PnC</i> in order to inform the EV emulator about the expected authorization method. -The test case calls <i>SendInstallISO15118CertificateMethod</i> method in order to trigger the EV emulator to initiate installing a new certificate.</p> <p>Memory State: <i>CertificateInstalled</i> from certificateType <i>V2GRootCertificate</i> <i>CertificateInstalled</i> from certificateType <i>MORootCertificate</i></p> <p>Reusable State(s): State is <i>EVConnectedPreSession</i></p> |

| Main (Test scenario) | |
|---|---|
| Charging Station | CSMS |
| 1. The Charging Station sends a Get15118EVCertificateRequest | 2. The Test System responds with a Get15118EVCertificateResponse with status <i>Accepted</i> exiResponse <i><Raw CertificateInstallationRes response for the EV, Base64 encoded.></i> |
| 3. The Charging Station sends an AuthorizeRequest | 4. The Test System responds with an AuthorizeResponse with status <i>Accepted</i> |

| |
|---|
| Tool validations |
| <p>* Step 1: Message: Get15118EVCertificateRequest - action must be <i>Update</i></p> <p>Post scenario validations: N/a</p> |

TC_M_29_CS: Certificate Update EV - Failed

| | |
|-------------------|---|
| Test case name | Certificate Update EV - Failed |
| Test case Id | TC_M_29_CS |
| Use case Id(s) | M02 |
| Requirement(s) | M02.FR.01 |
| System under test | Charging Station |
| Description | The EV initiates updating the existing certificate. The Charging Station forwards the update request to the CSMS. |
| Purpose | To verify if the Charging Station is able to forward the request to the CSMS. |
| Prerequisite(s) | The Charging Station supports ISO 15118-2. |

Before (Preparations)

Configuration State:

ISO15118Ctrlr.ContractCertificateInstallationEnabled is *true*

-The test case calls *SendISO15118AuthorizationMethod* method with parameter *PnC* in order to inform the EV emulator about the expected authorization method.

-The test case calls *SendInstallISO15118CertificateMethod* method in order to trigger the EV emulator to initiate installing a new certificate.

Memory State:

CertificateInstalled from certificateType *V2GRootCertificate*

CertificateInstalled from certificateType *MORootCertificate*

Reusable State(s):

State is *EVConnectedPreSession*

Main (Test scenario)

| Charging Station | CSMS |
|---|--|
| 1. The Charging Station sends a Get15118EVCertificateRequest | 2. The Test System responds with a Get15118EVCertificateResponse with status Failed exiResponse is omitted. |
| | 3. If an AuthorizeRequest is received, the testcase will FAIL and the Test System reports why it failed. |

Tool validations

* Step 1:

Message: **Get15118EVCertificateRequest**

- **action** must be *Update*

Post scenario validations:

N/a

N Diagnostics

TC_N_01_CS: Get Monitoring Report - with monitoringCriteria

| | |
|-------------------|--|
| Test case name | Get Monitoring Report - with monitoringCriteria |
| Test case Id | TC_N_01_CS |
| Use case Id(s) | N02 |
| Requirement(s) | N02.FR.01, N02.FR.03,N02.FR.04, N02.FR.05, N02.FR.06 , N02.FR.09, N02.FR.12 , N02.FR.13 , N02.FR.14 |
| System under test | Charging Station |
| Description | CSMS requests a report of all monitors that match the given monitoringCriteria : Threshold, Delta or Periodic. |
| Purpose | To test that Charging Station supports reporting of monitoring via monitoringCriteria . Starting with ThresholdMonitoring and then extending the set to check that combinations are handled properly. |
| Prerequisite(s) | Charging Station has implemented device model monitoring and MonitoringCtrlr::Enabled = true. |

Before (Preparations)

Configuration State:

The following monitors must have been configured by CSMS for Component Variable <Configured threshold monitor component variable>:

- LowerThreshold using value <Configured threshold monitor component variable LowerThreshold trigger value>
- UpperThreshold using value <Configured threshold monitor component variable UpperThreshold trigger value>
- Periodic using value <Configured Clock Aligned MeterValues Interval>

+ The following monitors must have been configured by CSMS for Component Variable <Configured numeric delta component variable>:

- Delta using value <Configured numeric delta component variable Delta numeric trigger value>
- PeriodicClockAligned using value <Configured Clock Aligned MeterValues Interval>

Memory State:

N/a

Reusable State(s):

N/a

Main (Test scenario)

| Charging Station | CSMS |
|---|---|
| 2. Charging Station responds with: GetMonitoringReportResponse | 1. Test System sends GetMonitoringReportRequest with: - requestId = <Generated requestId> - monitoringCriteria = { <i>ThresholdMonitoring</i> } |
| 3. Charging Station responds with: NotifyMonitoringReportRequest | 4. Test System sends NotifyMonitoringReportResponse |
| Step 3 and 4 are repeated as often as needed to report all configuration variables. | |
| 6. Charging Station responds with: GetMonitoringReportResponse | 5. Test System sends GetMonitoringReportRequest with: - requestId = <Generated requestId> - monitoringCriteria = { <i>ThresholdMonitoring</i> , <i>DeltaMonitoring</i> } |
| 7. Charging Station responds with: NotifyMonitoringReportRequest | 8. Test System sends NotifyMonitoringReportResponse |
| Step 7 and 8 are repeated as often as needed to report all configuration variables. | |
| 10. Charging Station responds with: GetMonitoringReportResponse | 9. Test System sends GetMonitoringReportRequest with: - requestId = <Generated requestId> - monitoringCriteria = { <i>DeltaMonitoring</i> , <i>PeriodicMonitoring</i> } |
| 11. Charging Station responds with: NotifyMonitoringReportRequest | 12. Test System sends NotifyMonitoringReportResponse |
| Step 11 and 12 are repeated as often as needed to report all configuration variables. | |

Tool validations

* Step 2:

Message: **GetMonitoringReportResponse**

- **status** = *Accepted*
- **statusInfo** is absent or **statusInfo.reasonCode** = *"NoError"*

* Step 3:

Message: **NotifyMonitoringReportRequest**

- **requestId** = *<Generated requestId>*
- **generatedAt** = *<timestamp at charging station>*
- **seqNo** = *0*
- **monitor.variableMonitoring.type** = *UpperThreshold* or *LowerThreshold*

While **tbc** = *true*

Message: **NotifyMonitoringReportRequest**

- **seqNo** is incremented by 1
- **monitor.variableMonitoring.type** = *UpperThreshold* or *LowerThreshold*

* Step 6:

Message: **GetMonitoringReportResponse**

- **status** = *Accepted*
- **statusInfo** is absent or **statusInfo.reasonCode** = *"NoError"*

* Step 7:

Message: **NotifyMonitoringReportRequest**

- **requestId** = *<Generated requestId>*
- **generatedAt** = *<timestamp at charging station>*
- **seqNo** = *0*
- **monitor.variableMonitoring.type** = *UpperThreshold*, *LowerThreshold* or *Delta*

While **tbc** = *true*

Message: **NotifyMonitoringReportRequest**

- **seqNo** is incremented by 1
- **monitor.variableMonitoring.type** = *UpperThreshold*, *LowerThreshold* or *Delta*

* Step 10:

Message: **GetMonitoringReportResponse**

- **status** = *Accepted*
- **statusInfo** is absent or **statusInfo.reasonCode** = *"NoError"*

* Step 11:

Message: **NotifyMonitoringReportRequest**

- **requestId** = *<Generated requestId>*
- **generatedAt** = *<timestamp at charging station>*
- **seqNo** = *0*
- **monitor.variableMonitoring.type** = *Delta*, *Periodic* or *PeriodicClockAligned*

While **tbc** = *true*

Message: **NotifyMonitoringReportRequest**

- **seqNo** is incremented by 1
- **monitor.variableMonitoring.type** = *Delta*, *Periodic* or *PeriodicClockAligned*

Post scenario validations:

N/A

TC_N_02_CS: Get Monitoring Report - with component/variable

| | |
|-------------------|--|
| Test case name | Get Monitoring Report - with component/variable |
| Test case Id | TC_N_02_CS |
| Use case Id(s) | N02 |
| Requirement(s) | N02.FR.01, N02.FR.03, N02.FR.04, N02.FR.05, N02.FR.08 , N02.FR.09 |
| System under test | Charging Station |
| Description | CSMS requests a report of monitors that match the given list of components and variables. |
| Purpose | To test that Charging Station supports reporting of monitoring via for a given list of components and optionally with variables. |
| Prerequisite(s) | Charging Station has implemented device model monitoring and MonitoringCtrlr::Enabled = true. |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a <i>Note: these are required variables for which a monitor can be expected to exist or it can be configured.</i> |
| Memory State: The following monitors must be present as 'hard-wired' or 'preconfigured' or must have been configured by CSMS: - Component "ChargingStation", variable "AvailabilityState", monitor type <i>Delta</i> - Component "EVSE", <Configured evseId>, variable "AvailabilityState", monitor type <i>Delta</i> |
| Reusable State(s): N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. Charging Station responds with: GetMonitoringReportResponse | 1. Test System sends GetMonitoringReportRequest with: - requestId = <Generated requestId> - monitoringCriteria is omitted - componentVariable[0].component.name = "ChargingStation" - componentVariable[0].variable.name = "AvailabilityState" - componentVariable[1].component.name = "EVSE" - componentVariable[1].component.evse.id = <Configured evseId> Note: requesting AvailabilityState from ChargingStation and all monitors from Configured EVSE |
| 3. Charging Station responds with: NotifyMonitoringReportRequest | 4. Test System sends NotifyMonitoringReportResponse |
| Step 3 and 4 are repeated as often as needed to report all configuration variables. | |

| |
|---|
| Tool validations |
| * Step 2: Message: GetMonitoringReportResponse - status = <i>Accepted</i> - statusInfo is absent or statusInfo.reasonCode = "NoError" |
| * Step 3: Message: NotifyMonitoringReportRequest - requestId = <Generated requestId> - generatedAt = <timestamp at charging station> - seqNo = 0 - if monitor.variable = "AvailabilityState" then monitor.variableMonitoring.type = <i>Delta</i> Note: fore EVSE #1 we request all monitors. There may be other monitors besides AvailabilityState. |

| Tool validations | |
|---|--|
| While tbc = <i>true</i> | Message: NotifyMonitoringReportRequest <ul style="list-style-type: none">- seqNo is incremented by 1- monitor.variable = <i>"AvailabilityState"</i>- monitor.variableMonitoring.type = <i>Delta</i>- monitor.component_.name = <i>ChargingStation</i> or <i>EVSE</i> |
| Post scenario validations: <p>Check that a monitor for AvailabilityState of type <i>Delta</i> is reported for both ChargingStation and COnfigured EVSE. If other monitors are present on Configured EVSE, then they will also be reported.</p> | |

TC_N_03_CS: Get Monitoring Report - with component criteria and component/variable

| | |
|-------------------|--|
| Test case name | Get Monitoring Report - with component criteria and component/variable |
| Test case Id | TC_N_03_CS |
| Use case Id(s) | N02 |
| Requirement(s) | N02.FR.01, N02.FR.03,N02.FR.04, N02.FR.05 , N02.FR.09, N02.FR.10 , N02.FR.13 |
| System under test | Charging Station |
| Description | CSMS requests a report of monitors that match both the component criteria and the given list of components and variables. |
| Purpose | To test that Charging Station supports reporting of monitoring for both the component criteria and a given list of components and optionally with variables. |
| Prerequisite(s) | Charging Station has implemented device model monitoring and MonitoringCtrlr::Enabled = true. |

Before (Preparations)

Configuration State:

The following monitors must be present as 'hard-wired' or 'preconfigured' or must have been configured by CSMS:

- Component "ChargingStation", variable "Power", monitor type *Periodic*
- Component "EVSE", evse <Configured evseld>, variable "AvailabilityState", monitor type *Delta*

Note: these are required variables for which a monitor can be expected to exist or it can be configured.

Memory State:

N/a

Reusable State(s):

N/a

Main (Test scenario)

| Charging Station | CSMS |
|--|--|
| 2. Charging Station responds with: GetMonitoringReportResponse | 1. Test System sends GetMonitoringReportRequest with: <ul style="list-style-type: none"> - requestId = <Generated requestId1> - monitoringCriteria is <i>ThresholdMonitoring</i> - componentVariable[0].component.name = "ChargingStation" - componentVariable[0].variable.name = "AvailabilityState" - componentVariable[1].component.name = "EVSE" - componentVariable[1].component.evse.id = <Configured evseld> - componentVariable[1].variable.name = "AvailabilityState" <i>Note: requesting AvailabilityState from ChargingStation and Configured EVSE, but filtered to _ThresholdMonitoring._</i> |
| 4. Charging Station responds with: GetMonitoringReportResponse | 3. Test System sends GetMonitoringReportRequest with: <ul style="list-style-type: none"> - requestId = <Generated requestId2> - monitoringCriteria is <i>DeltaMonitoring</i> - componentVariable[0].component.name = "ChargingStation" - componentVariable[0].variable.name = "AvailabilityState" - componentVariable[1].component.name = "EVSE" - componentVariable[1].component.evse.id = <Configured evseld> - componentVariable[1].variable.name = "AvailabilityState" <i>Note: requesting AvailabilityState from ChargingStation and Configured EVSE, but filtered to _Delta._</i> |
| 5. Charging Station responds with: NotifyMonitoringReportRequest | 6. Test System sends NotifyMonitoringReportResponse |
| <i>Step 5 and 6 are repeated as often as needed to report all configuration variables.</i> | |

| Tool validations |
|--|
| <p>* Step 2:</p> <p>Message: GetMonitoringReportResponse</p> <ul style="list-style-type: none">- status = <i>EmptyResultSet</i>- statusInfo is absent or statusInfo.reasonCode = <i>"NotFound"</i> <p>* Step 4:</p> <p>Message: GetMonitoringReportResponse</p> <ul style="list-style-type: none">- status = <i>Accepted</i>- statusInfo is absent or statusInfo.reasonCode = <i>"NoError"</i> <p>* Step 5:</p> <p>Message: NotifyMonitoringReportRequest</p> <ul style="list-style-type: none">- requestId = <i><Generated requestId></i>- generatedAt = <i><timestamp at charging station></i>- seqNo = <i>0</i>- monitor.variableMonitoring.type = <i>Delta</i> <p>While tbc = <i>true</i></p> <p>Message: NotifyMonitoringReportRequest</p> <ul style="list-style-type: none">- seqNo is incremented by 1- monitor.variableMonitoring.type = <i>Delta</i> |
| <p>Post scenario validations:</p> <p>Check that nothing is reported for requestId = <i><Generated requestId1></i> and a monitor for AvailabilityState of type <i>Delta</i> is reported for both ChargingStation and EVSE #1 for requestId = <i><Generated requestId2></i>.</p> |

TC_N_04_CS: Get Monitoring Report - for unknown component criteria

NOTE

Since MonitoringCriterionEnum is defined as enumeration, this will most likely already be caught by the JSON parser.

| | |
|-------------------|---|
| Test case name | Get Monitoring Report - for unknown component criteria |
| Test case Id | TC_N_04_CS |
| Use case Id(s) | N02 |
| Requirement(s) | N02.FR.02 |
| System under test | Charging Station |
| Description | CSMS sends a GetMonitoringReport with an invalid value in monitoringCriteria . |
| Purpose | To test that Charging Station returns a <i>NotSupported</i> return code in response to an invalid value for monitoringCriteria . |
| Prerequisite(s) | Charging Station has implemented device model monitoring and MonitoringCtrlr::Enabled = true. |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|--|---|
| Charging Station | CSMS |
| 1. Charging Station responds with: GetMonitoringReportResponse | 2. Test System sends GetMonitoringReportRequest with: - requestId = <Generated requestId> - monitoringCriteria = { <i>DeltaMonitoring</i> , <Configured <i>Unsupported monitoringCriteria</i> > } - *componentVariable is absent |

| |
|---|
| Tool validations |
| * Step 1 Message: GetMonitoringReportResponse - status = <i>NotSupported</i> - statusInfo is absent or statusInfo.reasonCode = " <i>UnsupportedParam</i> " or statusInfo.reasonCode = " <i>InvalidValue</i> " |
| Post scenario validations: N/A |

TC_N_05_CS: Set Monitoring Base - success

| | |
|-------------------|---|
| Test case name | Set Monitoring Base - success |
| Test case Id | TC_N_05_CS |
| Use case Id(s) | N03 |
| Requirement(s) | N03.FR.01, N03.FR.03, N03.FR.04, N03.FR.05 |
| System under test | Charging Station |
| Description | CSMS sends a SetMonitoringBaseRequest for All, FactoryDefault and HardWiredOnly. |
| Purpose | To test that Charging Station supports all three monitoring base types. |
| Prerequisite(s) | Charging Station has implemented device model monitoring and MonitoringCtrlr::Enabled = true. |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State(s):

N/a

Main (Test scenario)

| Charging Station | CSMS |
|--|---|
| 2. Charging Station responds with: SetMonitoringBaseResponse | 1. Test System sends SetMonitoringBaseRequest with: - monitoringBase = <i>All</i> |
| 4. Charging Station responds with: SetMonitoringBaseResponse | 3. Test System sends SetMonitoringBaseRequest with: - monitoringBase = <i>FactoryDefault</i> |
| 6. Charging Station responds with: SetMonitoringBaseResponse | 5. Test System sends SetMonitoringBaseRequest with: - monitoringBase = <i>HardWiredOnly</i> |

Tool validations

* Step 2

Message: **SetMonitoringBaseResponse**

- **status** = *Accepted*
- **statusInfo** is absent or **statusInfo.reasonCode** = *"NoError"*

* Step 4

Message: **SetMonitoringBaseResponse**

- **status** = *Accepted*
- **statusInfo** is absent or **statusInfo.reasonCode** = *"NoError"*

* Step 6

Message: **SetMonitoringBaseResponse**

- **status** = *Accepted*
- **statusInfo** is absent or **statusInfo.reasonCode** = *"NoError"*

Post scenario validations:

N/A

TC_N_06_CS: Set Monitoring Base - test removal custom monitors

| | |
|--------------------------|---|
| Test case name | Set Monitoring Base - test removal custom monitors |
| Test case Id | TC_N_06_CS |
| Use case Id(s) | N03 |
| Requirement(s) | N03.FR.01, N03.FR.05 |
| System under test | Charging Station |
| Description | CSMS sends a SetMonitoringBaseRequest for HardWiredOnly. |
| Purpose | To test that Charging Station removes custom monitors when selecting a monitoring base, as specified explicitly in N03.FR.05 and less formally in the remark of the use case N03. |
| Prerequisite(s) | Charging Station has implemented device model monitoring and MonitoringCtrlr::Enabled = true. |

| |
|---|
| Before (Preparations) |
| Configuration State: The following monitor must be present as 'preconfigured' or custom monitor configured by CSMS: - Component "ChargingStation", variable "AvailabilityState", monitor type <i>Delta</i> If it exists as a hardwired monitor, then the test will fail, because the test checks that it is removed when reverting back to only hardwired monitors. <i>Note: this is a required variable for which a monitor can be expected to exist or it can be configured.</i> |
| Memory State: N/a |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|--|--|
| Charging Station | CSMS |
| Check that monitor AvailabilityState exists. | |
| 2. Charging Station responds with: GetMonitoringReportResponse | 1. Test System sends GetMonitoringReportRequest with: - requestId = <Generated requestId> - monitoringCriteria is absent - componentVariable[0].component.name = "ChargingStation" - componentVariable[0].variable.name = "AvailabilityState" |
| 3. Charging Station responds with: NotifyMonitoringReportRequest | 4. Test System sends NotifyMonitoringReportResponse |
| 6. Charging Station responds with: SetMonitoringBaseResponse | 5. Test System sends SetMonitoringBaseRequest with: - monitoringBase = HardWiredOnly |
| Check that monitor AvailabilityState has been removed. | |
| 8. Charging Station responds with: GetMonitoringReportResponse | 7. Test System sends GetMonitoringReportRequest with: - requestId = <Generated requestId> - monitoringCriteria is absent - componentVariable[0].component.name = "ChargingStation" - componentVariable[0].variable.name = "AvailabilityState" |

| |
|--|
| Tool validations |
| * Step 2 Message: GetMonitoringReportResponse - status = Accepted - statusInfo is absent or statusInfo.reasonCode = "NoError" |

| Tool validations |
|---|
| <p>* Step 3:</p> <p>Message: NotifyMonitoringReportRequest</p> <ul style="list-style-type: none">- requestId = <i><Generated requestId></i>- generatedAt = <i><timestamp at charging station></i>- seqNo = 0- tbc is absent or tbc = <i>false</i>- monitor.variableMonitoring.type = <i>Delta</i>- monitor.component.name = <i>"ChargingStation"</i>- monitor.variable.name = <i>"AvailabilityState"</i> |
| <p>* Step 6</p> <p>Message: SetMonitoringBaseResponse</p> <ul style="list-style-type: none">- status = <i>Accepted</i>- statusInfo is absent or statusInfo.reasonCode = <i>"NoError"</i> |
| <p>* Step 8</p> <p>Message: GetMonitoringReportResponse</p> <ul style="list-style-type: none">- status = <i>EmptyResultSet</i>- statusInfo is absent or statusInfo.reasonCode = <i>"NotFound"</i> |
| <p>Post scenario validations:</p> <p>N/A</p> |

TC_N_07_CS: Set Monitoring Base - for unknown base type

| | |
|------|---|
| NOTE | Since MonitoringBaseEnumType is defined as enumeration, this will most likely already be caught by the JSON parser. |
|------|---|

| | |
|-------------------|---|
| Test case name | Set Monitoring Base - for unknown base type |
| Test case Id | TC_N_07_CS |
| Use case Id(s) | N03 |
| Requirement(s) | N03.FR.02 |
| System under test | Charging Station |
| Description | CSMS send a SetMonitoringBase with an invalid value in monitoringBase . |
| Purpose | To test that Charging Station returns a <i>NotSupported</i> return code in response to an invalid value for monitoringBase . |
| Prerequisite(s) | Charging Station has implemented device model monitoring and MonitoringCtrlr::Enabled = true. |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. Charging Station responds with: SetMonitoringBaseResponse | 1. Test System sends SetMonitoringBaseRequest with: - monitoringBase = <Configured unsupported_monitoringBase> |

| |
|---|
| Tool validations |
| * Step 2 Message: SetMonitoringBaseResponse - status = NotSupported - statusInfo is absent or statusInfo.reasonCode = "UnsupportedParam" or statusInfo.reasonCode = "InvalidValue" |
| Post scenario validations: N/A |

TC_N_08_CS: Set Variable Monitoring - one setMonitoringData element

| | |
|-------------------|---|
| Test case name | Set Variable Monitoring - one setMonitoringData element |
| Test case Id | TC_N_08_CS |
| Use case Id(s) | N04 |
| Requirement(s) | N04.FR.01, N04.FR.02, N04.FR.08, N04.FR.11 |
| System under test | Charging Station |
| Description | CSMS sends a request to activate a monitor on a single variable. |
| Purpose | To test that Charging Station supports setting of a monitor on a variable. |
| Prerequisite(s) | Charging Station has implemented device model monitoring and MonitoringCtrlr::Enabled = true. |

Before (Preparations)

Configuration State:

This test case activates a monitor on the following variable:

- Component "EVSE", evse "1", variable "AvailabilityState", monitor type *Delta*

It assumes, that no monitor is active on this variable prior to the test.

Note: this is a required variable for which a monitor can be expected to exist or it can be configured.

Note 2: Any other component/variable combination that supports monitoring could also be used for this test case.

Memory State:

N/a

Reusable State(s):

N/a

Main (Test scenario)

| Charging Station | CSMS |
|--|--|
| 2. Charging Station responds with: SetVariableMonitoringResponse | <i>Install monitor</i> 1. Test System sends SetVariableMonitoringRequest with: - setMonitoringData.value = 1 - setMonitoringData.type = <i>Delta</i> - setMonitoringData.severity = 8 - setMonitoringData.component.name = "EVSE" - setMonitoringData.component.evse.id = <Configured evseld> - setMonitoringData.variable.name = "AvailabilityState" |
| 4. Charging Station responds with: GetMonitoringReportResponse | <i>Verify monitor is installed</i> 3. Test System sends GetMonitoringReportRequest with: - requestId = <Generated requestId> - monitoringCriteria is absent - componentVariable[0].component.name = "EVSE" - componentVariable[0].component.evse.id = <Configured evseld> - componentVariable[0].variable.name = "AvailabilityState" |
| 5. Charging Station responds with: NotifyMonitoringReportRequest | 6. Test System sends NotifyMonitoringReportResponse |

| Tool validations |
|---|
| <p>* Step 2:</p> <p>Message: SetVariableMonitoringResponse with:</p> <p>setMonitoringResult = {</p> <ul style="list-style-type: none">- status = <i>Accepted</i>- type = <i>Delta</i>- severity = <i>8</i>- component.name = <i>"EVSE"</i>- component.evse.id = <i><Configured evseld></i>- variable.name = <i>"AvailabilityState"</i>- statusInfo is absent or statusInfo.reasonCode = <i>"NoError"</i> <p>}</p> |
| <p>* Step 4:</p> <p>Message: GetMonitoringReportResponse</p> <ul style="list-style-type: none">- status = <i>Accepted</i>- statusInfo is absent or statusInfo.reasonCode = <i>"NoError"</i> |
| <p>* Step 5:</p> <p>Message: NotifyMonitoringReportRequest</p> <ul style="list-style-type: none">- requestId = <i><Generated requestId></i>- monitor.variableMonitoring.type = <i>Delta</i>- monitor.component.name = <i>"EVSE"</i>- monitor.component.evse.id = <i><Configured evseld></i>- monitor.variable.name = <i>"AvailabilityState"</i> |
| <p>Post scenario validations:</p> |

TC_N_09_CS: Set Variable Monitoring - Multiple elements on different component and variable

| | |
|-------------------|---|
| Test case name | Set Variable Monitoring - Multiple elements on different component and variable |
| Test case Id | TC_N_09_CS |
| Use case Id(s) | N04 |
| Requirement(s) | N04.FR.01, N04.FR.02, N04.FR.08, N04.FR.11 |
| System under test | Charging Station |
| Description | CSMS sends a request to activate monitors on different variables. |
| Purpose | To test that Charging Station supports setting of multiple monitors on different variables. |
| Prerequisite(s) | Charging Station has implemented device model monitoring and MonitoringCtrlr::Enabled = true. |

| |
|---|
| Before (Preparations) |
| <p>Configuration State:</p> <p>This test case activates monitors on the following variables:</p> <ul style="list-style-type: none"> - Component "EVSE", evse "1", variable "AvailabilityState", monitor type <i>Delta</i> - Component "ChargingStation", variable "AvailabilityState", monitor type <i>Delta</i> <p>It assumes, that no monitor is active on these variables prior to the test.</p> <p><i>Note: these are required variables for which a monitor can be expected to exist or it can be configured.</i></p> <p><i>Note 2: Any other component/variable combination that supports monitoring could also be used for this test case.</i></p> <p>Memory State: N/a</p> <p>Reusable State(s): N/a</p> |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <p>2. Charging Station responds with: SetVariableMonitoringResponse</p> | <p><i>Install monitors</i></p> <p>1. Test System sends SetVariableMonitoringRequest with:</p> <ul style="list-style-type: none"> - setMonitoringData[0].value = 1 - setMonitoringData[0].type = <i>Delta</i> - setMonitoringData[0].severity = <i><Configured severity></i> - setMonitoringData[0].component.name = "EVSE" - setMonitoringData[0].component.evse.id = <i><Configured evseId></i> - setMonitoringData[0].variable.name = "AvailabilityState" - setMonitoringData[1].value = 1 - setMonitoringData[1].type = <i>Delta</i> - setMonitoringData[1].severity = <i><Configured severity></i> - setMonitoringData[1].component.name = "ChargingStation" - setMonitoringData[1].variable.name = "AvailabilityState" |
| <p>4. Charging Station responds with: GetMonitoringReportResponse</p> | <p><i>Verify monitors are installed</i></p> <p>3. Test System sends GetMonitoringReportRequest with:</p> <ul style="list-style-type: none"> - requestId = <i><Generated requestId></i> - monitoringCriteria is absent - componentVariable[0].component.name = "EVSE" - - componentVariable[0].component.evse.id = 1 - componentVariable[0].variable.name = "AvailabilityState" - componentVariable[1].component.name = "ChargingStation" - componentVariable[1].variable.name = "AvailabilityState" |

| Main (Test scenario) | |
|---|--|
| 5. Charging Station responds with: NotifyMonitoringReportRequest | 6. Test System sends NotifyMonitoringReportResponse |
| Step 5 and 6 may be repeated if the result is not sent in one report message. | |

| Tool validations | |
|---|---|
| <p>* Step 2:</p> <p>Message: SetVariableMonitoringResponse with:</p> <p>setMonitoringResult[1]</p> <ul style="list-style-type: none"> - id = <id of new monitor> - status = <i>Accepted</i> - type = <i>Delta</i> - severity = 8 - component.name = "EVSE" - component.evse.id = <Configured evseId> - variable.name = "AvailabilityState" - statusInfo is absent or statusInfo.reasonCode = "NoError" <p>setMonitoringResult[2]</p> <ul style="list-style-type: none"> - id = <id of new monitor> - status = <i>Accepted</i> - type = <i>Delta</i> - severity = 8 - component.name = "ChargingStation" - variable.name = "AvailabilityState" - statusInfo is absent or statusInfo.reasonCode = "NoError" <p>* Step 4:</p> <p>Message: GetMonitoringReportResponse</p> <ul style="list-style-type: none"> - status = <i>Accepted</i> - statusInfo is absent or statusInfo.reasonCode = "NoError" * Step 5: <p>Message: NotifyMonitoringReportRequest</p> <ul style="list-style-type: none"> - requestId = <Generated requestId> - generatedAt = <timestamp at charging station> - seqNo = 0 | |
| while tb is true | <p>Expect NotifyMonitoringReportRequest</p> <ul style="list-style-type: none"> - seqNo is incremented by 1 |
| <p>Post scenario validations:</p> <p>Verify the following monitors are reported in arbitrary order:</p> <p>monitor[1] = {</p> <ul style="list-style-type: none"> - variableMonitoring.type = <i>Delta</i> - variableMonitoring.severity = 8 - monitor.component.name = "EVSE" - monitor.component.evse.id = 1 - monitor.variable.name = "AvailabilityState" } <p>monitor[2] = {</p> <ul style="list-style-type: none"> - variableMonitoring.type = <i>Delta</i> - variableMonitoring.severity = 8 - monitor.component.name = "ChargingStation" - monitor.variable.name = "AvailabilityState" } | |

TC_N_10_CS: Set Variable Monitoring - Multiple monitors on the same component and variable

| | |
|-------------------|--|
| Test case name | Set Variable Monitoring - Multiple monitors on the same component and variable |
| Test case Id | TC_N_10_CS |
| Use case Id(s) | N04 |
| Requirement(s) | N04.FR.01, N04.FR.02, N04.FR.08, N04.FR.11 |
| System under test | Charging Station |
| Description | CSMS sets multiple monitors on the same component/variable combination. |
| Purpose | To test that Charging Station supports multiple monitors on same component/variable combination. |
| Prerequisite(s) | Charging Station has implemented device model monitoring and MonitoringCtrlr::Enabled = true. |

| |
|---|
| <p>Before (Preparations)</p> <p>Configuration State: This test case activates two monitors on the following variable: - Component "EVSE", evse "1", variable "AvailabilityState", monitor type <i>Delta</i></p> <p><i>Note: it does not make any practical sense to install two <i>_Delta</i> monitors on same variable with different severity, because a <i>Delta</i> monitor on a non-numeric variable is triggered by any change. However, the specification allows for it, therefore we use this variable, because it must exist.</i> If the variable "Power" can be monitored on an EVSE, then it is much more realistic to use that with a combination of two different UpperThresholds and severities._</p> <p>Memory State: N/a</p> <p>Reusable State(s): N/a</p> |
|---|

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <p>2. Charging Station responds with: SetVariableMonitoringResponse</p> | <p><i>Install monitors</i></p> <p>1. Test System sends SetVariableMonitoringRequest with:</p> <ul style="list-style-type: none"> - setMonitoringData[0].value = 1 - setMonitoringData[0].type = <i>Delta</i> - setMonitoringData[0].severity = 8 - setMonitoringData[0].component.name = "EVSE" - setMonitoringData[0].component.evse.id = <Configured evseId> - setMonitoringData[0].variable.name = "AvailabilityState" - setMonitoringData[1].value = 1 - setMonitoringData[1].type = <i>Delta</i> - setMonitoringData[1].severity = 7 - setMonitoringData[1].component.name = "EVSE" - setMonitoringData[1].component.evse.id = <Configured evseId> - setMonitoringData[1].variable.name = "AvailabilityState" |

| Tool validations |
|--|
| <p>* Step 2:</p> <p>Message: SetVariableMonitoringResponse with (in arbitrary order):</p> <p>setMonitoringResult[1] = {</p> <ul style="list-style-type: none">- id = <id of new monitor>- status = <i>Accepted</i>- type = <i>Delta</i>- severity = 8- component.name = "EVSE"- component.evse.id = <Configured evseld>- variable.name = "AvailabilityState"- statusInfo is absent or statusInfo.reasonCode = "NoError" <p>}</p> <p>setMonitoringResult[2] = {</p> <ul style="list-style-type: none">- id = <id of new monitor>- status = <i>Accepted</i>- type = <i>Delta</i>- severity = 7- component.name = "EVSE"- component.evse.id = <Configured evseld>- variable.name = "AvailabilityState"- statusInfo is absent or statusInfo.reasonCode = "NoError" <p>}</p> |
| <p>Post scenario validations:</p> <p>N/A</p> |

TC_N_11_CS: Set Variable Monitoring - Unknown component

| | |
|-------------------|---|
| Test case name | Set Variable Monitoring - Unknown component |
| Test case Id | TC_N_11_CS |
| Use case Id(s) | N04 |
| Requirement(s) | N04.FR.03 |
| System under test | Charging Station |
| Description | CSMS tries to set a monitor on an unknown component. |
| Purpose | To test that Charging Station checks whether a component exists. |
| Prerequisite(s) | Charging Station has implemented device model monitoring and MonitoringCtrlr::Enabled = true. |

| |
|---|
| Before (Preparations) |
| <p>Configuration State: This test case activates a monitor on an existing component on non-existing evse and then on a non-existing component "NonExistent":</p> <ul style="list-style-type: none">- Component "EVSE", evse "99", variable "AvailabilityState", monitor type <i>Delta</i>- Component "NonExistent", variable "Power", monitor type <i>UpperThreshold</i> <p><i>Note: this assumes, that EVSE #99 does not exist. The response to the "NonExistent" component can be either UnknownComponent or UnknownVariable, because both will not exist.</i></p> |
| <p>Memory State: N/a</p> |
| <p>Reusable State(s): N/a</p> |

| Main (Test scenario) | |
|--|--|
| Charging Station | CSMS |
| 2. Charging Station responds with: SetVariableMonitoringResponse | <i>Install monitors</i> 1. Test System sends SetVariableMonitoringRequest with: setMonitoringData[1] = { - setMonitoringData[0].value = 1 - setMonitoringData[0].type = Delta - setMonitoringData[0].severity = <Configured severity> - setMonitoringData[0].component.name = "EVSE" - setMonitoringData[0].component.evse.id = 99 - setMonitoringData[0].variable.name = "AvailabilityState" } setMonitoringData[2] = { - setMonitoringData[1].value = 1234.0 - setMonitoringData[1].type = UpperThreshold - setMonitoringData[1].severity = <Configured severity> - setMonitoringData[1].component.name = "NonExistent" - setMonitoringData[1].variable.name = "Power" } |

| Tool validations |
|---|
| <p>* Step 2:</p> <p>Message: SetVariableMonitoringResponse with (in arbitrary order):</p> <ul style="list-style-type: none">- id is absent- status = <i>UnknownComponent</i> or <i>Rejected</i>- type = <i>Delta</i>- severity = <i><Configured severity></i>- component.name = <i>"EVSE"</i>- component.evse.id = 99- variable.name = <i>"AvailabilityState"</i>- statusInfo is absent or statusInfo.reasonCode = <i>"UnknownEVSE"</i> or statusInfo = <i>"NotFound"</i>- id is absent- status = <i>UnknownComponent</i> (<i>UnknownVariable</i> will also be allowed, but is less accurate)- type = <i>UpperThreshold</i>- severity = <i><Configured severity></i>- component.name = <i>"NonExistent"</i>- variable.name = <i>"AvailabilityState"</i>- statusInfo is absent or statusInfo.reasonCode = <i>"NotFound"</i> |
| <p>Post scenario validations:</p> <p>N/A</p> |

TC_N_12_CS: Set Variable Monitoring - Value out of range - Delta monitor

| | |
|-------------------|---|
| Test case name | Set Variable Monitoring - Value out of range - Delta monitor |
| Test case Id | TC_N_12_CS |
| Use case Id(s) | N04 |
| Requirement(s) | N04.FR.14 |
| System under test | Charging Station |
| Description | CSMS tries to set a delta monitor with a value that is out of range. |
| Purpose | To test that Charging Station checks that value is within range of variable. |
| Prerequisite(s) | Charging Station has implemented device model monitoring and MonitoringCtrlr::Enabled = true. This test case assumes a numeric component variable exists which can be monitored. |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|--|---|
| Charging Station | CSMS |
| 2. Charging Station responds with: SetVariableMonitoringResponse | <i>Install monitors</i> 1. Test System sends SetVariableMonitoringRequest with: - setMonitoringData[0].value = -1 - setMonitoringData[0].type = <i>Delta</i> - setMonitoringData[0].severity = <i><Configured severity></i> - setMonitoringData[0].component = <i><Configured numeric delta component></i> - setMonitoringData[0].variable = <i><Configured numeric delta component variable></i> |

| |
|--|
| Tool validations |
| * Step 2: Message: SetVariableMonitoringResponse with (in arbitrary order): setMonitoringResult = { - id is absent - status = <i>Rejected</i> - type = <i>Delta</i> - severity = <i><Configured severity></i> - component = <i><Configured numeric delta component variable></i> - variable = <i><Configured numeric delta component variable></i> - statusInfo is absent or statusInfo.reasonCode = <i>"ValueOutOfRange"</i> or statusInfo.reasonCode = <i>"ValuePositiveOnly"</i> } Post scenario validations: N/A |

TC_N_13_CS: Set Variable Monitoring - Value out of range - Threshold monitor

| | |
|-------------------|--|
| Test case name | Set Variable Monitoring - Value out of range - Threshold monitor |
| Test case Id | TC_N_13_CS |
| Use case Id(s) | N04 |
| Requirement(s) | N04.FR.06 |
| System under test | Charging Station |
| Description | CSMS tries to set a threshold monitor with a value that is out of range. |
| Purpose | To test that Charging Station checks that value is within range of variable. |
| Prerequisite(s) | Charging Station has implemented device model monitoring and MonitoringCtrlr::Enabled = true. This test case assumes the <Configured threshold monitor component variable with maxLimit> component.variable exists and can be monitored and has variableCharacteristics.maxLimit < <Configured threshold monitor component variable with maxLimit exceeding maxLimit value> + Note: Variable _Power(maxLimit) is mandatory for an EVSE, but the actual value not, but it is likely (but not guaranteed), that it can be monitored._ |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. Charging Station responds with: SetVariableMonitoringResponse | <i>Install monitors</i> 1. Test System sends SetVariableMonitoringRequest with: <ul style="list-style-type: none"> - setMonitoringData[0].value = <Configured threshold monitor component variable with maxLimit exceeding maxLimit value> - setMonitoringData[0].type = UpperThreshold - setMonitoringData[0].severity = <Configured severity> - setMonitoringData[0].component = <Configured threshold monitor component variable with maxLimit> - setMonitoringData[0].variable = <Configured threshold monitor component variable with maxLimit> |

| |
|--|
| Tool validations |
| <p>* Step 2:</p> <p>Message: SetVariableMonitoringResponse with (in arbitrary order):</p> <pre>setMonitoringResult = { - id is absent - status = Rejected - type = UpperThreshold - severity = <Configured severity> - component = <Configured threshold monitor component variable with maxLimit> - variable = <Configured threshold monitor component variable with maxLimit> - statusInfo is absent or statusInfo.reasonCode = "ValueOutOfRange" }</pre> |
| Post scenario validations: N/A |

TC_N_15_CS: Set Variable Monitoring - Duplicate Variable type/severity combination

| | |
|-------------------|--|
| Test case name | Set Variable Monitoring - Duplicate Variable type/severity combination |
| Test case Id | TC_N_15_CS |
| Use case Id(s) | N04 |
| Requirement(s) | N04.FR.10 |
| System under test | Charging Station |
| Description | CSMS sets multiple monitors on the same component/variable combination with same severity and type. |
| Purpose | To test that Charging Station rejects multiple monitors on same component/variable combination when having the same severity and type. |
| Prerequisite(s) | Charging Station has implemented device model monitoring and MonitoringCtrlr::Enabled = true. |

| |
|---|
| Before (Preparations) |
| <p>Configuration State:</p> <p>This test case activates two monitors on the following variable:</p> <ul style="list-style-type: none"> - Component "EVSE", evse "1", variable "AvailabilityState", monitor type <i>Delta</i> + Note: it does not make any practical sense to install two <i>_Delta</i> monitors on same variable with different severity, because a <i>Delta</i> monitor on a non-numeric variable is triggered by any change. However, the specification allows for it, therefore we use this variable, because it must exist. <p>If the variable "Power" can be monitored on an EVSE, then it is much more realistic to use that with a combination of two different UpperThresholds and severities._</p> <p>Memory State:</p> <p>N/a</p> <p>Reusable State(s):</p> <p>N/a</p> |

| Main (Test scenario) | |
|--|---|
| Charging Station | CSMS |
| 2. Charging Station responds with: SetVariableMonitoringResponse | <i>Install monitors with same severity and of type _Delta</i> 1. Test System sends SetVariableMonitoringRequest with: <ul style="list-style-type: none">- setMonitoringData[0].value = 1- setMonitoringData[0].type = <i>Delta</i>- setMonitoringData[0].severity = <i><Configured severity></i>- setMonitoringData[0].component.name = "EVSE"- setMonitoringData[0].component.evse.id = <i><Configured evseId></i>- setMonitoringData[0].variable.name = "AvailabilityState"- setMonitoringData[1].value = 1- setMonitoringData[1].type = <i>Delta</i>- setMonitoringData[1].severity = <i><Configured severity></i>- setMonitoringData[1].component.name = "EVSE"- setMonitoringData[1].component.evse.id = <i><Configured evseId></i>- setMonitoringData[1].variable.name = "AvailabilityState" |

| Tool validations |
|---|
| <p>* Step 2:</p> <p>Message: SetVariableMonitoringResponse with (in arbitrary order):</p> <p>setMonitoringResult[1] = {</p> <ul style="list-style-type: none">- id = <i><id of new monitor></i>- status = <i>Accepted</i>- type = <i>Delta</i>- severity = <i><Configured severity></i>- component.name = <i>"EVSE"</i>- component.evse.id = <i><Configured evseld></i>- variable.name = <i>"AvailabilityState"</i>- statusInfo is absent or statusInfo.reasonCode = <i>"NoError"</i> <p>}</p> <p>setMonitoringResult[2] = {</p> <ul style="list-style-type: none">- status = <i>Duplicate</i>- type = <i>Delta</i>- severity = <i><Configured severity></i>- component.name = <i>"EVSE"</i>- component.evse.id = <i><Configured evseld></i>- variable.name = <i>"AvailabilityState"</i>- statusInfo is absent or statusInfo.reasonCode = <i>"InvalidValue"</i> <p>}</p> |
| <p>Post scenario validations:</p> <p>N/A</p> |

TC_N_16_CS: Set Monitoring Level - Success

| | |
|-------------------|--|
| Test case name | Set Monitoring Level - Success |
| Test case Id | TC_N_16_CS |
| Use case Id(s) | N05 |
| Requirement(s) | N05.FR.01, N05.FR.03 |
| System under test | Charging Station |
| Description | CSMS sets a monitoring level after which only monitors with lower or equal level are reported. |
| Purpose | To test that Charging Station accepts monitoring message and correctly filters events. |
| Prerequisite(s) | Charging Station has implemented device model monitoring and MonitoringCtrlr::Enabled = true. |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: This test case activates a monitor on the following variable: - Component "EVSE", variable "AvailabilityState", monitor type <i>Delta</i> , severity 8 It assumes that no monitor is active on this variable at start of the test. |
| Reusable State(s): N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <i>Set a monitoring level that suppresses the notification</i> | |
| 2. Charging Station responds with: SetMonitoringLevelResponse | 1. Test System sends: SetMonitoringLevelRequest with: severity = 7 |
| 4 Charging Station does NOT send NotifyEventRequest for configured EVSE | 3. <i>Plugin cable to configured EVSE to make it _Occupied</i> and test that notification is suppressed._ |

| |
|--|
| Tool validations |
| * Step 2: Message: SetMonitoringLevelResponse with: status = Accepted statusInfo is absent or statusInfo.reasonCode = "NoError" |
| Post scenario validations: Verify that no event notification is sent for the configured EVSE. |

TC_N_17_CS: Set Monitoring Level - Out of range

| | |
|--------------------------|---|
| Test case name | Set Monitoring Level - Out of range |
| Test case Id | TC_N_17_CS |
| Use case Id(s) | N05 |
| Requirement(s) | N05.FR.02 |
| System under test | Charging Station |
| Description | CSMS sets a monitoring level with an out of range value. |
| Purpose | To test that Charging Station rejects monitoring message with out of range severity. |
| Prerequisite(s) | Charging Station has implemented device model monitoring and MonitoringCtrlr::Enabled = true. |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State(s):

N/a

Main (Test scenario)

| | |
|--|--|
| 2. Charging Station responds with: SetMonitoringLevelResponse | 1. Test System sends: SetMonitoringLevelRequest with: severity = 10 |
| 4. Charging Station responds with: SetMonitoringLevelResponse | 3. Test System sends: SetMonitoringLevelRequest with: severity = -1 |

Tool validations

* Step 2:

Message: **SetMonitoringLevelResponse** with:

- **status** = *Rejected*
- **statusInfo** is absent or **statusInfo.reasonCode** = "ValueOutOfRange" or **statusInfo.reasonCode** = "ValueTooHigh"

* Step 4:

Message: **SetMonitoringLevelResponse** with:

- **status** = *Rejected*
- **statusInfo** is absent or **statusInfo.reasonCode** = "ValueOutOfRange" or **statusInfo.reasonCode** = "ValueTooLow"

Post scenario validations:

N/A

TC_N_18_CS: Clear Monitoring - Success

| | |
|-------------------|---|
| Test case name | Clear Monitoring - Success |
| Test case Id | TC_N_18_CS |
| Use case Id(s) | N06 |
| Requirement(s) | N06.FR.01, N06.FR.05 |
| System under test | Charging Station |
| Description | CSMS clears a monitor that is identified by its id. |
| Purpose | To test that Charging Station clears the monitor. |
| Prerequisite(s) | Charging Station has implemented device model monitoring and MonitoringCtrlr::Enabled = true. |

Before (Preparations)

Configuration State:

N/a

Memory State:

- Component "ChargingStation", variable "AvailabilityState"
- Component "EVSE", variable "AvailabilityState"

Reusable State(s):

N/a

Main (Test scenario)

| | |
|--|--|
| 2. Charging Station responds with: ClearVariableMonitoringResponse | 1. Test System sends: ClearVariableMonitoringRequest with: - id = { ID1, ID2 } |
| 4. Charging Station responds with: GetMonitoringReportResponse | <i>Verify monitors are cleared</i> 3. Test System sends GetMonitoringReportRequest with: - requestId = <Generated requestId> - monitoringCriteria is absent - componentVariable[0].component.name = "ChargingStation" - componentVariable[0].variable.name = "AvailabilityState" - componentVariable[1].component.name = "EVSE" - componentVariable[1].component.evse.id = 1 - componentVariable[1].variable.name = "AvailabilityState" |

Tool validations

* Step 2:

Message: **ClearVariableMonitoringResponse** with (in arbitrary order):**clearMonitoringResult[1]:**

- status = *Accepted*
- id = <ID1>
- statusInfo is absent or statusInfo.reasonCode = "NoError"

clearMonitoringResult[2]:

- status = *Accepted*
- id = <ID2>
- statusInfo is absent or statusInfo.reasonCode = "NoError"

* Step 4:

Message: **GetMonitoringReportResponse** with:

- status = *EmptyResultSet*
- statusInfo is absent or statusInfo.reasonCode = "NotFound"

Post scenario validations:

N/A

TC_N_19_CS: Clear Monitoring - Not found

| | |
|-------------------|---|
| Test case name | Clear Monitoring - Not found |
| Test case Id | TC_N_19_CS |
| Use case Id(s) | N06 |
| Requirement(s) | N06.FR.02 |
| System under test | Charging Station |
| Description | CSMS clears a monitor that does not exist. |
| Purpose | To test that Charging Station responds with <i>NotFound</i> result. |
| Prerequisite(s) | Charging Station has implemented device model monitoring and MonitoringCtrlr::Enabled = true. |

| |
|--|
| Before (Preparations) |
| Configuration State: Test System Sends a GetMonitoringReportRequest, the CS then reports all existings monitors if it has any. If any monitors exist the tool will take the highest id number and add 1, if no monitors are reported a preconfigured number is used. |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|--|
| Main (Test scenario) | |
| This test uses a monitor id, that is expected not to exist. | |
| 2. Charging Station responds with: ClearVariableMonitoringResponse | 1. Test System sends: ClearVariableMonitoringRequest with: - <i>id</i> monitor id from the Preparations |

| |
|--|
| Tool validations |
| * Step 2: Message: ClearVariableMonitoringResponse with: clearMonitoringResult: - status = <i>NotFound</i> - id = 123456 - statusInfo is absent or statusInfo.reasonCode = "NotFound" |
| Post scenario validations: N/A |

TC_N_20_CS: Alert Event - Threshold value exceeded

| | |
|-------------------|---|
| Test case name | Alert Event - Threshold value exceeded |
| Test case Id | TC_N_20_CS |
| Use case Id(s) | N07 |
| Requirement(s) | N07.FR.06, N07.FR.07, N07.FR.16, N07.FR.17 |
| System under test | Charging Station |
| Description | A monitored variable exceeds a threshold monitor and causes a NotifyEventRequest message to be sent. |
| Purpose | To test that Charging Station supports threshold monitors |
| Prerequisite(s) | Charging Station has implemented device model monitoring and MonitoringCtrlr::Enabled = true. |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: This test requires the Monitoring Base to be set to <i>All</i> . - SetMonitoringBaseRequest with monitoringBase = <i>All</i> . Furthermore this test requires the existence of a <i>LowerThreshold</i> and <i>UpperThreshold</i> monitor on a (numerical) variable. Since it is not mandated which variables are required to be monitored, this test used the variable "Power" of component "EVSE". - setMonitoringData[0].value = <Configured threshold monitor component variable <i>UpperThreshold</i> trigger value> - setMonitoringData[0].type = <i>UpperThreshold</i> - setMonitoringData[0].severity = 5 - setMonitoringData[0].component = <Configured threshold monitor component variable> - setMonitoringData[0].variable = <Configured threshold monitor component variable> Set MonitoringLevel to 8 <u>Notes:</u> - Take a threshold that can easily be exceeded. |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>EnergyTransferStarted</i> or manually trigger the monitor. <u>Notes:</u> If componentVariable is set to "Power" or "Current" <i>EnergyTransferStarted</i> will trigger the monitor. If another componentvariable is chosen a manual action is needed to trigger the monitor. | |
| 2. Charging Station sends a NotifyEventRequest with: - Power exceeding upper threshold | 3. Test System responds with a NotifyEventResponse |
| 5. Charging Station responds with a SetVariableMonitoringResponse with: - status <i>Accepted</i> | 4. Test System sends a SetVariableMonitoringRequest with: - type <i>LowerThreshold</i> - component <Configured threshold monitor component variable> - variable <Configured threshold monitor component variable> - value <Configured threshold monitor component variable <i>LowerThreshold</i> trigger value> <u>Notes:</u> - Take a threshold that won't be exceeded. |
| 6. Execute Reusable State <i>StopAuthorized</i> or manually trigger the second monitor. <u>Notes:</u> If componentVariable is set to "Power" or "Current" <i>EnergyTransferStarted</i> will trigger the monitor. If another componentvariable is chosen a manual action is needed to trigger the monitor. | |
| 7. Charging Station sends: NotifyEventRequest for 2 events: - Returning below upper threshold (<i>cleared</i>) - Dropping below lower threshold | 8. Test System responds: NotifyEventResponse |

Main (Test scenario)

Notes: Steps 2, 3, 7, and 8 may be repeated if the data is sent using two requests instead of one.
Depending on the configuration the Charging Station may also send other notifications during step 4 and 9.

Tool validations

* Step 2: Message: **NotifyEventRequest** with:

- **generatedAt** = <time of generation at Charging Station>
 - **seqNo** = 0
- and an **eventData** element with:
- **eventId** = <id1>
 - **timestamp** = <time of event at Charging Station>
 - **trigger** = *Alerting*
 - **actualValue** = <current power> (must be > <Configured threshold monitor value>)
 - **cleared** is absent or **cleared** = *false*
 - **transactionId** = <transaction id> (delivery of power is always in transaction)
 - **variableMonitoringId** = <monitor id1>
 - **component** = <Configured threshold monitor component variable>
 - **variable.name** = <Configured threshold monitor component variable>

Other **eventData** elements can be ignored.

* Step 7: Message: **NotifyEventRequest** with:

- **generatedAt** = <time of generation at Charging Station>
 - **seqNo** = 0
- and an **eventData** element with:
- **eventId** = <id2>
 - **timestamp** = <time of event at Charging Station>
 - **trigger** = *Alerting*
 - **actualValue** = <current power> (must be =< <Configured threshold monitor value>)
 - **cleared** is true
 - **transactionId** = <transaction id> (delivery of power is always in transaction)
 - **variableMonitoringId** = <monitor id1>
 - **eventNotificationType** = *CustomMonitor*
 - **component** = <Configured threshold monitor component variable>
 - **variable.name** = <Configured threshold monitor component variable>

and an **eventData** element with:

- **eventId** = <id3>
- **timestamp** = <time of event at Charging Station>
- **trigger** = *Alerting*
- **actualValue** = <current power> (must be < <Configured threshold monitor2 value>)
- **cleared** is absent or **cleared** is false
- **transactionId** = <transaction id> (delivery of power is always in transaction)
- **variableMonitoringId** = <monitor id2>
- **eventNotificationType** = *CustomMonitor*
- **component** = <Configured threshold monitor component variable>
- **variable.name** = <Configured threshold monitor component variable>

Other **eventData** elements can be ignored. This can also be sent in two *NotifyEventRequests*, instead of one.

Post scenario validations:

N/A

TC_N_21_CS: Alert Event - Caused by hardwired trigger

| | |
|-------------------|---|
| Test case name | Alert Event - Caused by hardwired trigger |
| Test case Id | TC_N_21_CS |
| Use case Id(s) | N07 |
| Requirement(s) | |
| System under test | Charging Station |
| Description | An event that is hardwired in the firmware is reported. |
| Purpose | To test that Charging Station reports this as a <i>HardWiredNotification</i> . |
| Prerequisite(s) | Charging Station has implemented device model monitoring and MonitoringCtrlr::Enabled = true. This test assumes the existence of a hardwired notification in the Charging Station. The OCPP specification does not mandate any hardwired notifications, so it is up to the tester to select a certain notification and cause it to trigger the sending of an <i>NotifyEventRequest</i> . |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| Tester triggers Charging Station to send a hardwired notification for component _X and variable Y._ | |
| 1. Charging Station sends: NotifyEventRequest | 2. Test System responds: NotifyEventResponse |

| |
|---|
| Tool validations |
| <p>* Step 1: Message: NotifyEventRequest with:</p> <ul style="list-style-type: none"> - generatedAt = <time of generation at Charging Station> - seqNo = 0 <p>and an eventData element with:</p> <ul style="list-style-type: none"> - eventNotificationType = <i>HardWiredNotification</i> <p>Other eventData elements are not relevant for this test.</p> |
| Post scenario validations: N/A |

TC_N_22_CS: Offline Notification - OfflineMonitoringEventQueuingSeverity set equal or lower

| | |
|-------------------|---|
| Test case name | Offline Notification - OfflineMonitoringEventQueuingSeverity set equal or lower than severityLevel of the monitor |
| Test case Id | TC_N_22_CS |
| Use case Id(s) | N07 |
| Requirement(s) | N07.FR.04 |
| System under test | Charging Station |
| Description | Charging Station queues event notifications when offline. |
| Purpose | To test that Charging Station will queue event notifications with a severity equal or lower than OfflineMonitoringEventQueuingSeverity. |
| Prerequisite(s) | Charging Station is online at start of test for configuration. CS has implemented device model monitoring and MonitoringCtrlr::Enabled = true. |

| |
|--|
| Before (Preparations) |
| Configuration State: SetConfiguration with: - component.name = "MonitoringCtrlr" - variable.name = "OfflineQueuingSeverity" - attributeValue = <Configured Severity> |
| Memory State: Charging Station has a custom or predefined monitor on AvailabilityState for Configured EVSE with severity = <Configured severity> |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| Take Charging Station offline. | |
| 2. Charging Station queues event notification for EVSE #1::_AvailabilityState._ | 1. Plug a cable into EVSE #1 to generate an event notification for _AvailabilityState._ |
| <u>Note(s)</u> : The tool will now wait for <Configured Transaction Duration> seconds | |
| <u>Manual Action</u> : Bring Charging Station back online. | |
| 3. Charging Station sends NotifyEventRequest | 4. Test System responds with NotifyEventResponse |
| Steps 3 and 4 repeat for all queued events during the offline period | |

| |
|--|
| Tool validations |
| * Step 1: no communication |
| * Step 3: Validate that the following NotifyEventRequest message was received: with an eventData element with: - eventData[0].trigger = Delta - eventData[0].actualValue = "Occupied" - eventData[0].component.name = "EVSE" - eventData[0].component.evse.id = <Configured evseId> - eventData[0].variable.name = "AvailabilityState" |
| Post scenario validations: N/a |

TC_N_23_CS: Offline Notification - OfflineMonitoringEventQueuingSeverity set higher than severityLevel of the monitor

| | |
|-------------------|---|
| Test case name | Offline Notification - OfflineMonitoringEventQueuingSeverity set higher than severityLevel of the monitor |
| Test case Id | TC_N_23_CS |
| Use case Id(s) | N07 |
| Requirement(s) | N07.FR.04 |
| System under test | Charging Station |
| Description | Charging Station does not queue event notifications when offline. |
| Purpose | To test that Charging Station does not queue event notifications with a severity higher than OfflineMonitoringEventQueuingSeverity. |
| Prerequisite(s) | Charging Station is online at start of test for configuration. CS has implemented device model monitoring and MonitoringCtrlr::Enabled = true. |

| |
|--|
| Before (Preparations) |
| Configuration State: SetConfiguration with: - component.name = "MonitoringCtrlr" - variable.name = "OfflineQueuingSeverity" - attributeValue = <Configured Severity> |
| Memory State: Charging Station has custom or predefined monitors on variable AvailabilityState of Configured EVSE and Configured ConnectorId with severity = <Configured severity> + 1 |
| Reusable State(s): N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action:</u> Connect the EV and EVSE. | |
| 1. The Charging Station notifies the CSMS about the status change of the connector. | 2. The Test System responds accordingly. |
| <u>Note(s):</u> Step 3, 4, 5, 6, 7, and 8 need to be executed when TxStartPoint contains EVConnected OR ParkingBayOccupancy | |
| 3. The Charging Station sends a TransactionEventRequest | 4. The Test System responds with a TransactionEventResponse |
| <u>Manual Action:</u> Take Charging Station offline. | |
| <u>Manual Action:</u> Disconnect the EV and EVSE. | |
| <u>Manual Action:</u> Connect the EV and EVSE. | |
| <u>Note(s):</u> The tool will now wait for <Configured Transaction Duration> seconds | |
| <u>Manual Action:</u> Bring Charging Station back online. | |
| 5. The Charging Station sends a TransactionEventRequest | 6. The Test System responds with a TransactionEventResponse |
| 7. The Charging Station sends a TransactionEventRequest | 8. The Test System responds with a TransactionEventResponse |
| <u>Note(s):</u> The CS shall not send a NotifyEventRequest for AvailabilityState of EVSE and Connector. A StatusNotification may still be received. | |

| Tool validations |
|---|
| <p>* Step 1: <i>(Optional:)</i></p> <p>Message: StatusNotificationRequest</p> <ul style="list-style-type: none"> - evseld <configured evseld> - connectorId <configured connectorId> - connectorStatus must be <i>Occupied</i> <p><i>(Required, but can be combined into one NotifyEventRequest:)</i></p> <p>Message: NotifyEventRequest</p> <ul style="list-style-type: none"> - eventData[0].trigger must be <i>Delta</i> - eventData[0].actualValue must be <i>Occupied</i> - eventData[0].component.name must be <i>Connector</i> - eventData[0].component.evse.id must be <i>Configured EVSE</i> - eventData[0].component.evse.connectorId must be <i>Configured ConnectorId</i> - eventData[0].variable.name must be <i>AvailabilityState</i> <p>Message: NotifyEventRequest</p> <ul style="list-style-type: none"> - eventData[0].trigger must be <i>Delta</i> - eventData[0].actualValue must be <i>Occupied</i> - eventData[0].component.name must be <i>EVSE</i> - eventData[0].component.evse.id must be <i>Configured EVSE</i> - eventData[0].variable.name must be <i>AvailabilityState</i> |
| <p>* Step 3:</p> <p>Message: TransactionEventRequest</p> <ul style="list-style-type: none"> - triggerReason must be <i>CablePluggedIn</i> - transactionInfo.chargingState must be <i>EVConnected</i> |
| <p>* Step 5:</p> <p>Message: TransactionEventRequest</p> <ul style="list-style-type: none"> - triggerReason must be <i>EVCommunicationLost</i> - transactionInfo.chargingState must be <i>Idle</i> |
| <p>* Step 7:</p> <p>Message: TransactionEventRequest</p> <ul style="list-style-type: none"> - triggerReason must be <i>CablePluggedIn</i> - transactionInfo.chargingState must be <i>EVConnected</i> |
| <p>Post scenario validations:</p> <p>N/A</p> |

TC_N_24_CS: Set Variable Monitoring - Periodic event

| | |
|-------------------|---|
| Test case name | Set Variable Monitoring - Periodic event |
| Test case Id | TC_N_24_CS |
| Use case Id(s) | N04, N08 |
| Requirement(s) | N04.FR.01, N04.FR.08, N08.FR.05 and N08.FR.06 |
| System under test | Charging Station |
| Description | Charging Station sends a periodic event . |
| Purpose | To test that Charging Station sends periodic events |
| Prerequisite(s) | Charging Station has implemented device model monitoring and MonitoringCtrlr::Enabled = true. |

Before (Preparations)

Configuration State:

N/a

Memory State:

Set MonitoringLevel to 8

Reusable State(s):

N/a

Main (Test scenario)

| Charging Station | CSMS |
|---|--|
| Set the monitor to generate a periodic event notification | |
| 2. Charging Station responds with SetVariableMonitoringResponse | 1. Test System sends SetVariableMonitoringRequest with: <ul style="list-style-type: none"> - setMonitoringData[0].value = <Configured Clock Aligned MeterValues Interval> - setMonitoringData[0].type = Periodic - setMonitoringData[0].severity = 5 - setMonitoringData[0].component = <Configured threshold monitor component variable> - setMonitoringData[0].variable = <Configured threshold monitor component variable> |
| 3. Charging Station sends a NotifyEventRequest | 4. Test System responds with a NotifyEventResponse |
| <u>Note(s)</u> : Step 3 and 4 will repeat every <Configured Clock Aligned MeterValues Interval> seconds | |

Tool validations

* Step 2:

Message: **SetVariableMonitoringResponse** with:**setMonitoringResult[0].status** = Accepted**setMonitoringResult[0].type** = Periodic**setMonitoringResult[0].severity** = 5**setMonitoringResult[0].component** = <Configured threshold monitor component variable>**setMonitoringResult[0].variable** = <Configured threshold monitor component variable>**setMonitoringResult[0].attributeStatusInfo** is absent or **attributeStatusInfo.reasonCode** = "NoError"

* Step 3:

Message: **NotifyEventRequest** every <Configured Clock Aligned MeterValues Interval> seconds with:with an **eventData** element with:- **trigger** = Periodic- **component** = <Configured threshold monitor component variable>- **variable** = <Configured threshold monitor component variable>

Post scenario validations:

N/A

TC_N_25_CS: Retrieve Log Information - Diagnostics Log - Success

| | |
|-------------------|--|
| Test case name | Retrieve Log Information - Diagnostics Log - Success |
| Test case Id | TC_N_25_CS |
| Use case Id(s) | N01 |
| Requirement(s) | N01.FR.01, N01.FR.02, N01.FR.04, N01.FR.07, N01.FR.08, N01.FR.09, N01.FR.13 |
| System under test | Charging Station |
| Description | This test case covers the functionality of getting log information from a Charging Station. The CSMS can request a Charging Station to upload a file with log information to a given location (URL). The format of this log file is not prescribed. The Charging Station successfully uploads a log file and gives information about the status of the upload by sending status notifications to the CSMS. |
| Purpose | To verify if the Charging station is able to successfully upload a log as described at the OCPP specification. |
| Prerequisite(s) | <ul style="list-style-type: none"> - Charging Station has log information available. - A diagnostics logging server has been setup supporting one of the file transfer protocols supported by the Charging Station (This is configured at the configuration variable FileTransferProtocols). |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a GetLogResponse | 1. The Test System sends a GetLogRequest with logType <i>DiagnosticsLog</i> |
| <u>Note(s):</u> - Charging Station is uploading log file | |
| 3. The Charging Station sends a LogStatusNotificationRequest | 4. The Test System responds with a LogStatusNotificationResponse |
| <u>Note(s):</u> - Log file is uploaded | |
| 5. The Charging Station sends a LogStatusNotificationRequest | 6. The Test System responds with a LogStatusNotificationResponse |

| |
|--|
| Tool validations |
| * Step 2: Message GetLogResponse - status <i>Accepted</i> - filename <i>not omitted AND not empty</i> * Step 3: Message LogStatusNotificationRequest - status <i>Uploading</i> - requestId <i>Same Id as the GetLogRequest</i> * Step 5: Message LogStatusNotificationRequest - status <i>Uploaded</i> - requestId <i>Same Id as the GetLogRequest</i> |
| Post scenario validations: - N/a |

TC_N_26_CS: Retrieve Log Information - Diagnostics Log - Upload failed

| | |
|-------------------|--|
| Test case name | Retrieve Log Information - Diagnostics Log - Upload failed |
| Test case Id | TC_N_26_CS |
| Use case Id(s) | N01 |
| Requirement(s) | N01.FR.01, N01.FR.02, N01.FR.04, N01.FR.07, N01.FR.08, N01.FR.10, N01.FR.13 |
| System under test | Charging Station |
| Description | This test case covers the functionality of getting log information from a Charging Station. The CSMS can request a Charging Station to upload a file with log information to a given location (URL). The format of this log file is not prescribed. The Charging Station unsuccessfully uploads a log file and gives information about the status of the upload by sending status notifications to the CSMS. |
| Purpose | To verify if the Charging Station is able to correctly communicate with the CSMS after failing to upload a log as described at the OCPP specification. |
| Prerequisite(s) | - Charging Station has log information available. |

Before (Preparations)

Configuration State:

The retry interval should be configured longer than the time it takes to attempt an upload.

Memory State:

Charging Station has log information available.

Reusable State:

N/a

Main (Test scenario)

| Charging Station | CSMS |
|---|---|
| 2. The Charging Station responds with a GetLogResponse | 1. The Test System sends a GetLogRequest with - logType <i>DiagnosticsLog</i> - retries 3 - retryInterval <Configured retryInterval> - log.remoteLocation <Configured log location with a nonexistent path> |
| 3. The Charging Station sends a LogStatusNotificationRequest | 4. The Test System responds with a LogStatusNotificationResponse |
| 5. The Charging Station sends a LogStatusNotificationRequest | 6. The Test System responds with a LogStatusNotificationResponse |

Note(s):

- Step 3-4, 5-6 and 3-6 may repeat multiple times depending on Charging Station's implementation.
- The Test System waits at least (3 * <Configured retryInterval>), before ending the testcase.

| Tool validations |
|--|
| <div><div>* Step 2:</div><div>Message GetLogResponse</div><div><div>- status <i>Accepted</i></div></div><div>* Step 3:</div><div>Must be sent exactly 1 or 4 times</div><div>Message LogStatusNotificationRequest</div><div><div>- status <i>Uploading</i></div><div>- requestId <i>Same Id as the GetLogRequest</i></div></div><div>* Step 5:</div><div>Must be sent exactly 1 or 4 times</div><div>Message LogStatusNotificationRequest</div><div><div>- status <i>UploadFailure</i></div><div>- requestId <i>Same Id as the GetLogRequest</i></div></div><div>OR Message LogStatusNotificationRequest</div><div><div>- status <i>BadMessage</i></div><div>- requestId <i>Same Id as the GetLogRequest</i></div></div><div>OR Message LogStatusNotificationRequest</div><div><div>- status <i>PermissionDenied</i></div><div>- requestId <i>Same Id as the GetLogRequest</i></div></div><div>OR Message LogStatusNotificationRequest</div><div><div>- status <i>NotSupportedOperation</i></div><div>- requestId <i>Same Id as the GetLogRequest</i></div></div><div>* The time between the first LogStatusNotificationRequest <i>Uploading</i> and the last LogStatusNotificationRequest <i>UploadFailure/BadMessage/PermissionDenied/NotSupportedOperation</i> equals $(3 * \text{<Configured retryInterval>})$</div></div> |
| <div><div>Post scenario validations:</div><div><div>- N/a</div></div></div> |

TC_N_27_CS: Get Customer Information - Accepted + data

| | |
|-------------------|---|
| Test case name | Get Customer Information - Accepted + data |
| Test case Id | TC_N_27_CS |
| Use case Id(s) | N09 |
| Requirement(s) | N09.FR.02, N09.FR.05 |
| System under test | Charging Station |
| Description | The CSMS sends a message to the Charging Station to retrieve IdToken customer information, for example to be compliant with local privacy laws. The Charging Station notifies the CSMS by sending one or more reports. |
| Purpose | To verify if the Charging Station accepts the request and correctly sends the information as described at the OCPP specification. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The Charging Station needs to support Local Authorization and either the Local Authorization List or Authorization Cache. - The Charging Station supports authorization methods other than NoAuthorization |

Before (Preparations)

Configuration State:

LocalAuthListCtrlr.Enabled is set to *true*

AuthCtrlr.LocalPreAuthorize is set to *true*

AuthCacheCtrlr.Enabled is set to *true*

Memory State:

IdTokenCached for <Configured valid IdToken fields> (If implemented)

IdTokenLocalAuthList for <Configured valid IdToken fields> (If implemented)

Reusable State:

State is *Authorized* (local)

State is *ParkingBayUnoccupied*

Main (Test scenario)

| Charging Station | CSMS |
|--|---|
| 2. The Charging Station responds with a CustomerInformationResponse | 1. The Test System sends a CustomerInformationRequest with - report <i>true</i> - idToken <Configured valid idToken fields> |
| 3. The Charging Station sends a NotifyCustomerInformationRequest | 4. The Test System responds with a NotifyCustomerInformationResponse . |
| Note(s): - If <i>tbc</i> is <i>True</i> at Step 3 then step 3 and 4 will be repeated | |

Tool validations

* Step 2:

Message **CustomerInformationResponse**

- **status** *Accepted*

* Step 3:

Message **NotifyCustomerInformationRequest**

- **data** *Not empty*

Post scenario validations:

- All report parts have been received

TC_N_28_CS: Get Customer Information - Accepted + no data

| | |
|-------------------|--|
| Test case name | Get Customer Information - Accepted + no data |
| Test case Id | TC_N_28_CS |
| Use case Id(s) | N09 |
| Requirement(s) | N09.FR.02, N09.FR.06 |
| System under test | Charging Station |
| Description | The CSMS sends a message to the Charging Station to retrieve IdToken customer information, for example to be compliant with local privacy laws. The Charging Station notifies the CSMS by sending one or more reports. |
| Purpose | To verify if the Charging Station accepts the request and correctly respond when it couldn't find the right information as described at the OCPP specification. |
| Prerequisite(s) | Charging Station has no customer information available of <Configured valid idToken fields> |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: The CSMS requests the CS to clear the customerInformation for idToken <Configured valid idToken fields> |
| Reusable State: N/a |

| Main (Test scenario) | |
|--|--|
| Charging Station | CSMS |
| 2. The Charging Station responds with a CustomerInformationResponse | 1. The Test System sends a CustomerInformationRequest with - report <i>true</i> - idToken <Configured valid idToken fields> |
| 3. The Charging Station sends a NotifyCustomerInformationRequest | 4. The Test System responds with a NotifyCustomerInformationResponse . |

| |
|--|
| Tool validations |
| * Step 2: Message CustomerInformationResponse - status Accepted * Step 3: Message NotifyCustomerInformationRequest - tbv Not true |
| Post scenario validations: - A message is sent indicating that no data is found |

TC_N_29_CS: Get Customer Information - Not Accepted

| | |
|--------------------------|--|
| Test case name | Get Customer Information - Not Accepted |
| Test case Id | TC_N_29_CS |
| Use case Id(s) | N09 |
| Requirement(s) | N09.FR.03 |
| System under test | Charging Station |
| Description | The CSMS sends a message to the Charging Station to retrieve IdToken customer information, for example to be compliant with local privacy laws. The Charging Station notifies the CSMS by sending one or more reports. |
| Purpose | To verify if the Charging Station correctly responds when it cannot process the request as described at the OCPP specification. |
| Prerequisite(s) | Charging station is in a state where it cannot process customer information requests |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a CustomerInformationResponse | 1. The Test System sends a CustomerInformationRequest |

| |
|---|
| Tool validations |
| * Step 2: Message CustomerInformationResponse - status <i>Invalid</i> |
| Post scenario validations: - N/a |

TC_N_30_CS: Clear Customer Information - Clear and report + data

| | |
|-------------------|---|
| Test case name | Clear Customer Information - Clear and report + data |
| Test case Id | TC_N_30_CS |
| Use case Id(s) | N10 |
| Requirement(s) | N10.FR.01, N10.FR.03 |
| System under test | Charging Station |
| Description | The CSMS sends a message to the Charging Station to clear (and retrieve) IdToken customer information, for example to be compliant with local privacy laws. The Charging Station notifies the CSMS by sending one or more reports. |
| Purpose | To verify if the Charging Station accepts the request and removes all customer related data (except from LocalList) and sent notifies as described at the OCPP specification. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The Charging Station needs to support Local Authorization and either the Local Authorization List or Authorization Cache. - The Charging Station supports authorization methods other than NoAuthorization |

Before (Preparations)

Configuration State:

LocalAuthListCtrlr.Enabled is set to *true*

AuthCtrlr.LocalPreAuthorize is set to *true*

AuthCacheCtrlr.Enabled is set to *true*

Memory State:

IdTokenCached for <Configured valid IdToken fields> (If implemented)

IdTokenLocalAuthList for <Configured valid IdToken fields> (If implemented)

Reusable State:

State is *Authorized* (local)

State is *ParkingBayUnoccupied*

Main (Test scenario)

| Charging Station | CSMS |
|--|--|
| 2. The Charging Station responds with a CustomerInformationResponse | 1. The Test System sends a CustomerInformationRequest with <ul style="list-style-type: none"> - report <i>true</i> AND - clear <i>true</i> AND - idToken <Configured valid idToken fields> |
| 3. The Charging Station sends a NotifyCustomerInformationRequest | 4. The Test System responds with a NotifyCustomerInformationResponse . |
| <u>Note(s):</u> - If tbc is <i>True</i> at Step 3 then step 3 and 4 will be repeated | |
| 6. The Charging Station responds with a CustomerInformationResponse | 5. The Test System sends a CustomerInformationRequest with <ul style="list-style-type: none"> - report <i>true</i> AND - idToken <Configured valid idToken fields> |
| 7. The Charging Station sends a NotifyCustomerInformationRequest | 8. The Test System responds with a NotifyCustomerInformationResponse . |
| <u>Note(s):</u> - Step is optional and only expected when status is <i>Accepted</i> at Step 6 | |

| Tool validations |
|---|
| <div>* Step 2: Message CustomerInformationResponse - status <i>Accepted</i></div> <div>* Step 3: Message NotifyCustomerInformationRequest - data <i>Not empty</i></div> <div>* Step 8: Message NotifyCustomerInformationRequest - tbc <i>Not true</i></div> |
| <div>Post scenario validations: - All report parts have been received</div> |

TC_N_31_CS: Clear Customer Information - Clear and report + no data

| | |
|--------------------------|--|
| Test case name | Clear Customer Information - Clear and report + no data |
| Test case Id | TC_N_31_CS |
| Use case Id(s) | N10 |
| Requirement(s) | N10.FR.01, N10.FR.04 |
| System under test | Charging Station |
| Description | The CSMS sends a message to the Charging Station to clear (and retrieve) IdToken customer information, for example to be compliant with local privacy laws. The Charging Station notifies the CSMS by sending one or more reports. |
| Purpose | To verify if the Charging Station accepts the request and correctly respond when it couldn't find the right information as described at the OCPP specification. |
| Prerequisite(s) | Charging Station has no customer information available of <Configured valid idToken fields> |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a CustomerInformationResponse | 1. The Test System sends a CustomerInformationRequest with - report <i>true</i> AND - clear <i>true</i> AND - idToken <Configured valid idToken fields> |
| 3. The Charging Station sends a NotifyCustomerInformationRequest | 4. The Test System responds with a NotifyCustomerInformationResponse . |

| |
|--|
| Tool validations |
| * Step 2: Message CustomerInformationResponse - status <i>Accepted</i> |
| Post scenario validations: - A message is send indicating that no data is found |

TC_N_32_CS: Clear Customer Information - Clear and no report

| | |
|--------------------------|--|
| Test case name | Clear Customer Information - Clear and no report |
| Test case Id | TC_N_32_CS |
| Use case Id(s) | N10 |
| Requirement(s) | N10.FR.01, N10.FR.06 |
| System under test | Charging Station |
| Description | The CSMS sends a message to the Charging Station to clear (and retrieve) IdToken customer information, for example to be compliant with local privacy laws. The Charging Station notifies the CSMS by sending one or more reports. |
| Purpose | To verify if the Charging Station accepts the request and removes all customer related data (except from LocalList) and sent one notify as described at the OCPP specification. |
| Prerequisite(s) | Charging Station has no customer information available of <Configured valid idToken fields> |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a CustomerInformationResponse | 1. The Test System sends a CustomerInformationRequest with - report <i>false</i> AND - clear <i>true</i> AND - idToken <Configured valid idToken fields> |
| 3. The Charging Station sends a NotifyCustomerInformationRequest | 4. The Test System responds with a NotifyCustomerInformationResponse . |

| |
|--|
| Tool validations |
| * Step 2: Message CustomerInformationResponse - status <i>Accepted</i> |
| Post scenario validations: - A message is send indicating that the data is cleared |

TC_N_62_CS: Clear Customer Information - Clear and report - customerIdentifier

| | |
|-------------------|--|
| Test case name | Clear Customer Information - Clear and report - customerIdentifier |
| Test case Id | TC_N_62_CS |
| Use case Id(s) | N10 |
| Requirement(s) | N10.FR.01, N10.FR.03 |
| System under test | Charging Station |
| Description | The CSMS sends a message to the Charging Station to clear (and retrieve) raw customer information, for example to be compliant with local privacy laws. The Charging Station notifies the CSMS by sending one or more reports. |
| Purpose | To verify if the Charging Station accepts the request and removes all customer related data (except from LocalList) and sent notifies as described at the OCPP specification. |
| Prerequisite(s) | The Charging Station needs to support retrieving / deleting CustomerInformation - CustomerIdentifier. |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: The tester needs manually store the <Configured CustomerIdentifier> at the Charging Station. |
| Reusable State: N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a CustomerInformationResponse | 1. The Test System sends a CustomerInformationRequest with - report <i>true</i> AND - clear <i>true</i> AND - customerIdentifier <Configured customerIdentifier> |
| 3. The Charging Station sends a NotifyCustomerInformationRequest | 4. The Test System responds with a NotifyCustomerInformationResponse |
| <u>Note(s):</u> - If tbc is <i>True</i> at Step 3 then step 3 and 4 will be repeated | |
| 6. The Charging Station responds with a CustomerInformationResponse | 5. The Test System sends a CustomerInformationRequest with - report <i>true</i> AND - clear <i>false</i> AND - customerIdentifier <Configured customerIdentifier> |
| 7. The Charging Station sends a NotifyCustomerInformationRequest | 8. The Test System responds with a NotifyCustomerInformationResponse |
| <u>Note(s):</u> - If tbc is <i>True</i> at Step 7 then step 7 and 8 will be repeated | |

| |
|---|
| Tool validations |
| <p>* Step 2: Message CustomerInformationResponse - status <i>Accepted</i></p> <p>* Step 3: Message NotifyCustomerInformationRequest - data <i>Not empty</i></p> <p>* Step 6: Message CustomerInformationResponse - status <i>Accepted</i></p> <p>* Step 7: Message NotifyCustomerInformationRequest - data <i>empty</i></p> |
| <p>Post scenario validations: - All report parts have been received</p> |

TC_N_63_CS: Clear Customer Information - Clear and report - customerCertificate

| | |
|--------------------------|---|
| Test case name | Clear Customer Information - Clear and report - customerCertificate |
| Test case Id | TC_N_63_CS |
| Use case Id(s) | N10 |
| Requirement(s) | N10.FR.09 |
| System under test | Charging Station |
| Description | <p>The CSMS sends a message to the Charging Station to clear (and retrieve) customer certificate information, for example to be compliant with local privacy laws. The Charging Station notifies the CSMS by sending one or more reports.</p> <p>Note: The only customer certificate that could exist in a charging station is a PnC contract certificate, which should not remain in the charging station.</p> |
| Purpose | To verify if the Charging Station accepts the request and removes all customer related data and sent notifies as described at the OCPP specification. |
| Prerequisite(s) | The Charging Station needs to support retrieving / deleting CustomerInformation - CustomerCertificate. |

| |
|--|
| Before (Preparations) |
| <p>Configuration State: - AuthCtrlr.Enabled is <i>true</i> - AuthCtrlr.DisableRemoteAuthorization is <i>false</i> - ISO15118Ctrlr.CentralContractValidationAllowed is <i>false</i> - ISO15118Ctrlr.V2GCertificateInstallationEnabled is <i>true</i> - ISO15118Ctrlr.ContractCertificateInstallationEnabled is <i>true</i> - ISO15118Ctrlr.PnCEnabled is <i>true</i></p> |
| <p>Memory State: RenewV2GChargingStationCertificate (If none are present, when checking with GetInstalledCertificateIds.certificateType = V2GCertificateChain)</p> |
| <p>Reusable State: Execute Reusable State EVConnectedPreSession Execute Reusable State Authorized15118 (PnC) Execute Reusable State EnergyTransferStarted</p> |

| | |
|--|-------------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual action:</u> EV ends the charging session. <u>Note:</u> The Charging Station receives a SessionStopReq(Terminate) message from the EV to finish the transaction. | |

| Main (Test scenario) | |
|--|--|
| 2. The Charging Station responds with a CustomerInformationResponse | 1. The Test System sends a CustomerInformationRequest with - report <i>true</i> AND - clear <i>true</i> AND - customerCertificate <i>certificate hash data of contract certificate</i> |
| 3. The Charging Station sends a NotifyCustomerInformationRequest | 4. The Test System responds with a NotifyCustomerInformationResponse |
| <u>Note(s):</u> - If tbc is <i>True</i> at Step 3 then step 3 and 4 will be repeated | |
| 6. The Charging Station responds with a CustomerInformationResponse | 5. The Test System sends a CustomerInformationRequest with - report <i>true</i> AND - clear <i>false</i> AND - customerCertificate <i>certificate hash data of contract certificate</i> |
| 7. The Charging Station sends a NotifyCustomerInformationRequest | 8. The Test System responds with a NotifyCustomerInformationResponse |

| Tool validations |
|---|
| * Step 2: Message CustomerInformationResponse - status <i>Accepted</i> * Step 3: Message NotifyCustomerInformationRequest - data <i>empty</i> or <i>Not empty</i> if a customer certificate exists * Step 6: Message CustomerInformationResponse - status <i>Accepted</i> * Step 7: Message NotifyCustomerInformationRequest - data <i>empty</i> |
| Post scenario validations: - All report parts have been received |

TC_N_33_CS: Clear Customer Information - Invalid

| | |
|--------------------------|--|
| Test case name | Clear Customer Information - Invalid |
| Test case Id | TC_N_33_CS |
| Use case Id(s) | N10 |
| Requirement(s) | N10.FR.01, N10.FR.05 |
| System under test | Charging Station |
| Description | The CSMS sends a message to the Charging Station to clear (and retrieve) IdToken customer information, for example to be compliant with local privacy laws. The Charging Station notifies the CSMS by sending one or more reports. |
| Purpose | To verify if the Charging Station rejects the request when it cannot process as described at the OCPP specification. |
| Prerequisite(s) | Charging station is in a state where it cannot process customer information requests |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a CustomerInformationResponse | 1. The Test System sends a CustomerInformationRequest |

| |
|---|
| Tool validations |
| * Step 2: Message CustomerInformationResponse - status <i>Invalid</i> |
| Post scenario validations: - N/a |

TC_N_34_CS: Retrieve Log Information - Rejected

| | |
|-------------------|--|
| Test case name | Retrieve Log Information - Rejected |
| Test case Id | TC_N_34_CS |
| Use case Id(s) | N01 |
| Requirement(s) | N01.FR.05 |
| System under test | Charging Station |
| Description | This test case covers the functionality of getting log information from a Charging Station. The CSMS can request a Charging Station to upload a file with log information to a given location (URL). The format of this log file is not prescribed. The Charging Station successfully uploads a log file and gives information about the status of the upload by sending status notifications to the CSMS. |
| Purpose | To verify if the Charging station is able to reject the request when no information is available as described at the OCPP specification. |
| Prerequisite(s) | This testcase can only be executed if it is possible to have no log information available at the Charging Station. |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a GetLogResponse | 1. The Test System sends a GetLogRequest with logType <Configured logType> |

| |
|---|
| Tool validations |
| * Step 2: Message GetLogResponse - status <i>Rejected</i> |
| Post scenario validations: - N/a |

TC_N_35_CS: Retrieve Log Information - Security Log - Success

| | |
|-------------------|--|
| Test case name | Retrieve Log Information - Security Log - Success |
| Test case Id | TC_N_35_CS |
| Use case Id(s) | N01 |
| Requirement(s) | N01.FR.01, N01.FR.02, N01.FR.03, N01.FR.07, N01.FR.08, N01.FR.09, N01.FR.13 |
| System under test | Charging Station |
| Description | This test case covers the functionality of getting log information from a Charging Station. The CSMS can request a Charging Station to upload a file with log information to a given location (URL). The format of this log file is not prescribed. The Charging Station successfully uploads a log file and gives information about the status of the upload by sending status notifications to the CSMS. |
| Purpose | To verify if the Charging station is able to successfully upload a log as described at the OCPP specification. |
| Prerequisite(s) | N/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: Charging Station has log information available. |
| Reusable State: N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a GetLogResponse | 1. The Test System sends a GetLogRequest with logType <i>SecurityLog</i> |
| <u>Note(s):</u> - <i>Charging Station is uploading log file</i> | |
| 3. The Charging Station sends a LogStatusNotificationRequest | 4. The Test System responds with a LogStatusNotificationResponse . |
| <u>Note(s):</u> - <i>Log file is uploaded</i> | |
| 5. The Charging Station sends a LogStatusNotificationRequest | 6. The Test System responds with a LogStatusNotificationResponse . |

| |
|---|
| Tool validations |
| <p>* Step 2: Message GetLogResponse - status <i>Accepted</i></p> <p>* Step 3: Message LogStatusNotificationRequest - status <i>Uploading</i> - requestId <i>Same Id as the GetLogRequest</i></p> <p>* Step 5: Message LogStatusNotificationRequest - status <i>Uploaded</i> - requestId <i>Same Id as the GetLogRequest</i></p> |
| Post scenario validations: - N/a |

TC_N_36_CS: Retrieve Log Information - Second Request

| | |
|-------------------|--|
| Test case name | Retrieve Log Information - Second Request |
| Test case Id | TC_N_36_CS |
| Use case Id(s) | N01 |
| Requirement(s) | N01.FR.01, N01.FR.02, N01.FR.03, N01.FR.07, N01.FR.08, N01.FR.09, N01.FR.12, N01.FR.13 |
| System under test | Charging Station |
| Description | This test case covers the functionality of getting log information from a Charging Station. The CSMS can request a Charging Station to upload a file with log information to a given location (URL). The format of this log file is not prescribed. The Charging Station successfully uploads a log file and gives information about the status of the upload by sending status notifications to the CSMS. |
| Purpose | To verify if the Charging station is able to successfully start/cancel a upload on a second request as described at the OCPP specification. |
| Prerequisite(s) | The Charging Station supports cancelling an ongoing log file upload. |

Before (Preparations)

Configuration State:

N/a

Memory State:

Charging Station has log information available of <Configured logType>.

Reusable State:

N/a

Main (Test scenario)

| Charging Station | CSMS |
|---|--|
| 2. The Charging Station responds with a GetLogResponse | 1. The Test System sends a GetLogRequest with logType <Configured logType> and requestId <#1> |
| <u>Note(s):</u> - Charging Station is uploading log file | |
| 3. The Charging Station sends a LogStatusNotificationRequest | 4. The Test System responds with a LogStatusNotificationResponse . |
| <u>Note(s):</u> - Charging Station cancels uploading the first log file | |
| 6. The Charging Station responds with a GetLogResponse | 5. The Test System sends a GetLogRequest with logType <Configured logType> and requestId <#2> |
| Step 7 is allowed to occur before step 6 7. The Charging Station sends a LogStatusNotificationRequest | 8. The Test System responds with a LogStatusNotificationResponse . |
| <u>Note(s):</u> - Charging Station is uploading log file | |
| 9. The Charging Station sends a LogStatusNotificationRequest | 10. The Test System responds with a LogStatusNotificationResponse . |
| <u>Note(s):</u> - Log file is uploaded | |
| 11. The Charging Station sends a LogStatusNotificationRequest | 12. The Test System responds with a LogStatusNotificationResponse . |

| Tool validations |
|--|
| <p>* Step 2:</p> <p>Message GetLogResponse</p> <ul style="list-style-type: none">- status <i>Accepted</i> <p>* Step 3:</p> <p>Message LogStatusNotificationRequest</p> <ul style="list-style-type: none">- status <i>Uploading</i>- requestId <#1> |
| <p><i>Step 7 is allowed to occur before step 6</i></p> <p>* Step 6:</p> <p>Message GetLogResponse</p> <ul style="list-style-type: none">- status <i>AcceptedCanceled</i> <p>* Step 7:</p> <p>Message LogStatusNotificationRequest</p> <ul style="list-style-type: none">- status <i>AcceptedCanceled</i>- requestId <#1> |
| <p>* Step 9:</p> <p>Message LogStatusNotificationRequest</p> <ul style="list-style-type: none">- status <i>Uploading</i>- requestId <#2> <p>* Step 11:</p> <p>Message LogStatusNotificationRequest</p> <ul style="list-style-type: none">- status <i>Uploaded</i>- requestId <#2> |
| <p>Post scenario validations:</p> <ul style="list-style-type: none">- N/a |

TC_N_37_CS: Set Variable Monitoring - Unknown Variable

| | |
|-------------------|---|
| Test case name | Set Variable Monitoring - Unknown Variable |
| Test case Id | TC_N_37_CS |
| Use case Id(s) | N04 |
| Requirement(s) | N04.FR.04 |
| System under test | Charging Station |
| Description | This test case describes how the CSMS requests the Charging Station to set monitoring triggers on Variables. Multiple triggers can be set for upper or lower thresholds, delta changes or periodic reporting. |
| Purpose | To verify if the Charging station is able to correctly respond to the request when an unknown variable is sent as described at the OCPP specification. |
| Prerequisite(s) | Charging Station supports Monitoring |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a SetVariableMonitoringResponse | 1. The Test System sends a SetVariableMonitoringRequest with setMonitoringData.type <i>Delta</i> setMonitoringData.variable.name <i>unknownVariable</i> setMonitoringData.component.name <i>EVSE</i> |

| |
|---|
| Tool validations |
| <p>* Step 2:</p> <p>Message SetVariableMonitoringResponse</p> <ul style="list-style-type: none"> - setMonitoringResult[0].status <i>UnknownVariable</i> - setMonitoringResult[0].type <i>Delta</i> - setMonitoringResult[0].severity <i><Configured severity></i> - setMonitoringResult[0].component.name <i>EVSE</i> - setMonitoringResult[0].variable.name <i>unkownVariable</i> |
| <p>Post scenario validations:</p> <ul style="list-style-type: none"> - N/a |

TC_N_38_CS: Set Variable Monitoring - Not supported MonitorType

| | |
|-------------------|---|
| Test case name | Set Variable Monitoring - Not supported MonitorType |
| Test case Id | TC_N_38_CS |
| Use case Id(s) | N04 |
| Requirement(s) | N04.FR.05 |
| System under test | Charging Station |
| Description | This test case describes how the CSMS requests the Charging Station to set monitoring triggers on Variables. Multiple triggers can be set for upper or lower thresholds, delta changes or periodic reporting. |
| Purpose | To verify if the Charging station is able to correctly respond to the request when a not supported monitortype is sent as described at the OCPP specification. |
| Prerequisite(s) | <ul style="list-style-type: none"> - Charging Station supports Monitoring. - Charging station does not support one or more variableMonitoringTypes. |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| Main (Test scenario) | |
|--|---|
| Charging Station | CSMS |
| 2. The Charging Station responds with a SetVariableMonitoringResponse | 1. The Test System sends a SetVariableMonitoringRequest with setVariableData *setMonitoringData.type <i>UpperThreshold</i> setMonitoringData.variable.name <i>AvailabilityState</i> setMonitoringData.component.name <i>EVSE</i> |

| |
|---|
| Tool validations |
| * Step 2: Message SetVariableMonitoringResponse - setMonitoringResult[0].status <i>UnsupportedMonitorType</i> or Rejected - setMonitoringResult[0].type <i>UpperThreshold</i> - setMonitoringResult[0].component.name <i>EVSE</i> - setMonitoringResult[0].variable.name <i>AvailabilityState</i> |
| Post scenario validations: - N/a |

TC_N_39_CS: Set Variable Monitoring - Component/Variable combination does NOT correspond

| | |
|--------------------------|--|
| Test case name | Set Variable Monitoring - Component/Variable combination does NOT correspond |
| Test case Id | TC_N_39_CS |
| Use case Id(s) | N04 |
| Requirement(s) | N04.FR.16 |
| System under test | Charging Station |
| Description | This test case describes how the CSMS requests the Charging Station to set monitoring triggers on Variables. Multiple triggers can be set for upper or lower thresholds, delta changes or periodic reporting. |
| Purpose | To verify if the Charging station is able to correctly respond to the request when a Component/Variable combination which does NOT correspond is sent as described at the OCPP specification. |
| Prerequisite(s) | Charging Station supports Monitoring This test case assumes the Charging Station has a non-numeric Component Variable <Configured non-numeric delta component variable> and numeric Component Variable <Configured threshold monitor component variable>. |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a SetVariableMonitoringResponse | 1. The Test System sends a SetVariableMonitoringRequest with setMonitoringData[0].type <i>Delta</i> setMonitoringData[0].value 1 setMonitoringData[0].component = <Configured non-numeric delta component variable> setMonitoringData[0].variable = <Configured non-numeric delta component variable> |
| 4. The Charging Station responds with a SetVariableMonitoringResponse | 3. The Test System sends a SetVariableMonitoringRequest with setMonitoringData[0].id <SetVariableMonitoringResponse.setMonitoringResult[0].id of step 2> setMonitoringData[0].type <i>Delta</i> setMonitoringData[0].value 1 setMonitoringData[0].component = <Configured numeric delta component variable> setMonitoringData[0].variable = <Configured numeric delta component variable> |
| 6. The Charging Station responds with a GetMonitoringReportResponse | 5. The Test System sends a GetMonitoringReportRequest with - requestId <Generated requestId> |
| 7. The Charging Station sends a NotifyMonitoringReportRequest | 8. The Test System responds with a NotifyMonitoringReportResponse . |
| Note(s): - If tbc is True at Step 7 then step 7 and 8 will be repeated | |

| Tool validations |
|---|
| <p>* Step 2:</p> <p>Message SetVariableMonitoringResponse</p> <ul style="list-style-type: none">- setMonitoringResult[0].id <i><not omitted></i>- setMonitoringResult[0].status <i>Accepted</i>- setMonitoringResult[0].type <i>Delta</i>- setMonitoringResult[0].component = <i><Configured non-numeric delta component variable></i>- setMonitoringResult[0].variable = <i><Configured non-numeric delta component variable></i> <p>* Step 4:</p> <p>Message SetVariableMonitoringResponse</p> <ul style="list-style-type: none">- setMonitoringResult[0].status <i>Rejected</i>- setMonitoringResult[0].type <i>Delta</i>- setMonitoringResult[0].component = <i><Configured numeric delta component variable></i>- setMonitoringResult[0].variable = <i><Configured numeric delta component variable></i> <p>* Step 6:</p> <p>Message GetMonitoringReportResponse</p> <ul style="list-style-type: none">- status <i>Accepted</i> <p>* Step 7:</p> <p>Message NotifyMonitoringReportRequest</p> <p>Must contain a monitor with</p> <ul style="list-style-type: none">- monitor[0].component = <i><Configured non-numeric delta component variable></i>- monitor[0].variable = <i><Configured non-numeric delta component variable></i>- monitor[0].variableMonitoring[0].id = <i><SetVariableMonitoringResponse.setMonitoringResult[0].id of step 2></i>- monitor[0].variableMonitoring[0].value = <i>1</i>- monitor[0].variableMonitoring[0].type = <i>Delta</i> <p>Post scenario validations:</p> <ul style="list-style-type: none">- All report parts have been received |

TC_N_40_CS: Set Variable Monitoring - Replace Variable Monitor

| | |
|--------------------------|---|
| Test case name | Set Variable Monitoring - Replace Variable Monitor |
| Test case Id | TC_N_40_CS |
| Use case Id(s) | N04 |
| Requirement(s) | N04.FR.12 |
| System under test | Charging Station |
| Description | This test case describes how the CSMS requests the Charging Station to set monitoring triggers on Variables. Multiple triggers can be set for upper or lower thresholds, delta changes or periodic reporting. |
| Purpose | To verify if the Charging station is able to correctly replace an existing variable monitor as described at the OCPP specification. |
| Prerequisite(s) | Charging Station supports Monitoring |

Before (Preparations)

Configuration State:

N/a

Memory State:

Variable monitor is set for *Delta* with severity 5

Reusable State:

N/a

Main (Test scenario)

| Charging Station | CSMS |
|--|--|
| 2. The Charging Station responds with a SetVariableMonitoringResponse | 1. The Test System sends a SetVariableMonitoringRequest with setMonitoringData.id <Generated variableMonitoringId> AND setMonitoringData.type <i>Delta</i> setMonitoringData.severity 4 |
| 4. The Charging Station responds with a GetMonitoringReportResponse | 3. The Test System sends a GetMonitoringReportRequest with - requestId <Generated requestId> - componentVariable.component.name <i>EVSE</i> - componentVariable.component.evse.id <i>evseId</i> - componentVariable.variable.name <i>AvailabilityState</i> - monitoringCriteria <i>DeltaMonitoring_</i> |
| 5. The Charging Station sends a NotifyMonitoringReportRequest | 6. The Test System responds with a NotifyMonitoringReportResponse . |

Tool validations

* Step 2:

Message **SetVariableMonitoringResponse**

- **setMonitoringResult[0].status** *Accepted*
- **setMonitoringResult[0].type** *Delta*
- **setMonitoringResult[0].component.name** *EVSE*
- **setMonitoringResult[0].variable.name** *AvailabilityState*

* Step 4:

Message **GetMonitoringReportResponse**

- **status** *Accepted*

* Step 5:

Message **NotifyMonitoringReportRequest**

- **monitor.component.name** *EVSE*
- **monitor.variable.name** *AvailabilityState*
- **monitor.variableMonitoring.severity** 4

Post scenario validations:

- All report parts have been received

TC_N_41_CS: Set Variable Monitoring - Return to FactoryDefault

| | |
|-------------------|--|
| Test case name | Set Variable Monitoring - Return to FactoryDefault |
| Test case Id | TC_N_41_CS |
| Use case Id(s) | N03 |
| Requirement(s) | N03.FR.04, N04.FR.15 |
| System under test | Charging Station |
| Description | This test case describes how the CSMS requests the Charging Station to overrule a preconfigured monitor by a custom monitor. When monitoringBase is set to FactoryDefault the preconfigured monitor must return. |
| Purpose | To verify if the Charging station is able to correctly restore monitors to FactoryDefault. |
| Prerequisite(s) | - Charging Station supports Monitoring - A preconfigured monitor exists with <i>id</i> <Configured preconfigured monitor id>. |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: MonitoringBase has been set to <i>FactoryDefault</i> |
| Reusable State: N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with GetMonitoringReportResponse | 1. The Test System sends GetMonitoringReportRequest with requestId = <Generated requestId> |
| 3. The Charging Station sends NotifyMonitoringReportRequest | 4. The Test System responds with NotifyMonitoringReportResponse |
| <u>Note(s):</u> - If NotifyMonitoringReportRequest.tbc is True in Step 3 then step 3 and 4 will be repeated | |
| Search NotifyMonitoringReportRequest.monitoringReportData of step 3 to get a monitoringReportData.monitor WHERE monitor.variableMonitoring.id is <Configured preconfigured monitor id> AS <preconfiguredMonitor> | |
| 6. The Charging Station responds with a SetVariableMonitoringResponse | 5. The Test System sends a SetVariableMonitoringRequest with setMonitoringData.id <preconfiguredMonitor.variableMonitoring.id> setMonitoringData.transaction <preconfiguredMonitor.variableMonitoring.transaction> setMonitoringData.value <preconfiguredMonitor.variableMonitoring.value> setMonitoringData.type <preconfiguredMonitor.variableMonitoring.type> setMonitoringData.severity IF <preconfiguredMonitor.variableMonitoring.severity> < 9 THEN <preconfiguredMonitor.variableMonitoring.severity> + 1 ELSE 5 END IF setMonitoringData.component <preconfiguredMonitor.component> setMonitoringData.variable <preconfiguredMonitor.variable> |

| Main (Test scenario) | |
|---|--|
| 8. The Charging Station responds with a GetMonitoringReportResponse | 7. The Test System sends a GetMonitoringReportRequest with - requestId <Generated requestId> - id <preconfiguredMonitor.variableMonitoring.id> - componentVariable.component <preconfiguredMonitor.component> - componentVariable.variable <preconfiguredMonitor.variable> |
| 9. The Charging Station sends a NotifyMonitoringReportRequest | 10. The Test System responds with a NotifyMonitoringReportResponse . |
| 12. The Charging Station responds with a SetMonitoringBaseResponse with - status <i>Accepted</i> | 11. The Test System sends a SetMonitoringBaseRequest with - monitoringBase <i>FactoryDefault</i> |
| 14. The Charging Station responds with a GetMonitoringReportResponse | 13. The Test System sends a GetMonitoringReportRequest with - requestId <Generated requestId> - id <preconfiguredMonitor.variableMonitoring.id> - componentVariable.component <preconfiguredMonitor.component> - componentVariable.variable <preconfiguredMonitor.variable> |
| 15. The Charging Station sends a NotifyMonitoringReportRequest | 16. The Test System responds with a NotifyMonitoringReportResponse . |

| Tool validations |
|---|
| <p>* Step 2: Message GetMonitoringReportResponse - status <i>Accepted</i></p> <p>* Step 6: Message SetVariableMonitoringResponse - setMonitoringResult[0].status <i>Accepted</i> - setMonitoringResult[0].type <i>Delta</i> - setMonitoringResult[0].component <preconfiguredMonitor.component> - setMonitoringResult[0].variable <preconfiguredMonitor.variable></p> <p>* Step 8: Message GetMonitoringReportResponse - status <i>Accepted</i></p> <p>* Step 9: Message NotifyMonitoringReportRequest Should contain monitor: - monitor.component <preconfiguredMonitor.component> - monitor.variable <preconfiguredMonitor.variable> - monitor.variableMonitoring.id <preconfiguredMonitor.variableMonitoring.id> - monitor.variableMonitoring.severity IF <preconfiguredMonitor.variableMonitoring.severity> < 9 THEN <preconfiguredMonitor.variableMonitoring.severity> + 1 ELSE 5 END IF</p> <p>* Step 15: Message NotifyMonitoringReportRequest Should contain monitor: - monitor.component <preconfiguredMonitor.component> - monitor.variable <preconfiguredMonitor.variable> - monitor.variableMonitoring.id <preconfiguredMonitor.variableMonitoring.id> - monitor.variableMonitoring.severity <preconfiguredMonitor.variableMonitoring.severity></p> |

| |
|--|
| Tool validations |
| Post scenario validations: - All report parts have been received |

TC_N_43_CS: Set Variable Monitoring - First SetMonitoringData and third SetMonitoringData are valid, but the second contains an out of range value

| | |
|-------------------|--|
| Test case name | Set Variable Monitoring - First SetMonitoringData and third SetMonitoringData are valid, but the second contains an out of range value |
| Test case Id | TC_N_43_CS |
| Use case Id(s) | N04 |
| Requirement(s) | N/a |
| System under test | Charging Station |
| Description | This test case describes how the CSMS requests the Charging Station to set monitoring triggers on Variables. Multiple triggers can be set for upper or lower thresholds, delta changes or periodic reporting. |
| Purpose | To verify if the Charging station is able to correctly respond when one of requested variable monitor data is out of range replace as described at the OCPP specification. |
| Prerequisite(s) | Charging Station supports Monitoring This test case assumes the Charging Station has a numeric Component Variable <Configured numeric delta component variable> and numeric Component Variable <Configured threshold monitor component variable>. |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State:

N/a

Main (Test scenario)

| Charging Station | CSMS |
|---|---|
| 2. The Charging Station responds with a SetVariableMonitoringResponse | <p>1. The Test System sends a SetVariableMonitoringRequest with</p> <ul style="list-style-type: none"> - setMonitoringData[0].component = <Configured threshold monitor component variable> - setMonitoringData[0].variable = <Configured threshold monitor component variable> - setMonitoringData[0].value = <Configured threshold monitor component variable UpperThreshold trigger value> - setMonitoringData[0].type = UpperThreshold <ul style="list-style-type: none"> - setMonitoringData[1].component = <Configured numeric delta component variable> - setMonitoringData[1].variable = <Configured numeric delta component variable> - setMonitoringData[1].value = -1.0 - setMonitoringData[1].type = Delta <ul style="list-style-type: none"> - setMonitoringData[2].component = <Configured threshold monitor component variable> - setMonitoringData[2].variable = <Configured threshold monitor component variable> - setMonitoringData[2].value = <Configured threshold monitor component variable LowerThreshold trigger value> - setMonitoringData[2].type = LowerThreshold |

| Tool validations |
|---|
| <p>* Step 2:</p> <p>Message: SetVariableMonitoringResponse with (in arbitrary order):</p> <p>setMonitoringResult[1] = {</p> <ul style="list-style-type: none">- status = <i>Accepted</i>- type = <i>UpperThreshold</i>- statusInfo is absent or statusInfo.reasonCode = <i>"NoError"</i> <p>}</p> <p>setMonitoringResult[2] = {</p> <ul style="list-style-type: none">- status = <i>Rejected</i>- type = <i>Delta</i> <p>}</p> <p>setMonitoringResult[3] = {</p> <ul style="list-style-type: none">- status = <i>Accepted</i>- type = <i>LowerThreshold</i>- statusInfo is absent or statusInfo.reasonCode = <i>"NoError"</i> <p>}</p> |
| <p>Post scenario validations:</p> <ul style="list-style-type: none">- N/a |

TC_N_44_CS: Clear Monitoring - Rejected

| | |
|-------------------|---|
| Test case name | Clear Monitoring - Rejected |
| Test case Id | TC_N_44_CS |
| Use case Id(s) | N06 |
| Requirement(s) | N06.FR.03 |
| System under test | Charging Station |
| Description | A monitoring setting can be cleared (removed) by sending a ClearVariableMonitoringRequest with the id of the monitoring setting. |
| Purpose | To verify if the Charging station is able to correctly respond on a request to clear a monitor that cannot be cleared as described at the OCPP specification. |
| Prerequisite(s) | Charging Station supports Monitoring, Charging Station has hard-coded monitor(s) |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: Test System Sends a GetMonitoringReportRequest, the CS then reports all existings monitors if it has any. These monitors should be hard-coded and the first Id is used fot the TC. |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a ClearVariableMonitoringResponse | 1. The Test System sends a ClearVariableMonitoringRequest with id monitor id from the Preparations |

| |
|---|
| Tool validations |
| * Step 2: Message ClearVariableMonitoringResponse - clearMonitoringResult[0].status <i>Rejected</i> |
| Post scenario validations: - N/a |

TC_N_45_CS: Alert Event - Delta value exceeded

| | |
|-------------------|---|
| Test case name | Alert Event - Delta value exceeded |
| Test case Id | TC_N_45_CS |
| Use case Id(s) | N07 |
| Requirement(s) | N07.FR.06, N07.FR.07, N07.FR.18, N07.FR.19 |
| System under test | Charging Station |
| Description | NotifyEventRequest reports every Component/Variable for which a VariableMonitoring setting was triggered. Only the VariableMonitoring settings that are responsible for triggering an event are included. |
| Purpose | To verify if the Charging station is correctly communicating when a delta value has exceeded as described at the OCPP specification. |
| Prerequisite(s) | n/a |

Before (Preparations)

Configuration State:

N/a

Memory State:

Variable monitor is configured with:

- **setMonitoringData.component** = <Configured numeric delta component variable>
- **setMonitoringData.variable** = <Configured numeric delta component variable Delta numeric trigger value>
- **setMonitoringData.value** = <Configured numeric delta component variable>
- **setMonitoringData.type** = Delta
- **setMonitoringData.severity** = 5

Set MonitoringLevel to 8

Notes: If componentVariable is set to "Power" or "Current", the value is set to 100.0

Reusable State:

N/a

Main (Test scenario)

| | |
|--|---|
| Charging Station | CSMS |
| <u>Manual Action:</u> If componentVariable is set to "Power" or "Current" EnergyTransferStarted will trigger the monitor. If another componentvariable is chosen a manual action is needed to trigger the monitor. | |
| 1. Execute Reusable State <i>EnergyTransferStarted</i> or manually trigger the monitor. | |
| 2. The Charging Station sends a NotifyEventRequest | 3. The Test System responds with a NotifyEventResponse . |
| <u>Note(s):</u> - If tbc is True at Step 2 then step 1 and 3 will be repeated | |

Tool validations

* Step 2:

Message **NotifyEventRequest**

- **eventData[0].trigger** Delta
- **eventData[0].component** <Configured numeric delta component variable>
- **eventData[0].variable** <Configured numeric delta component variable>
- **eventData[0].variableMonitoringId** <SetVariableMonitoringResponse.setMonitoringResult.id in preparation phase>

Post scenario validations:

- N/a

TC_N_47_CS: Get Monitoring report - Report all

| | |
|-------------------|---|
| Test case name | Get Monitoring report - Report all |
| Test case Id | TC_N_47_CS |
| Use case Id(s) | N02 |
| Requirement(s) | N02.FR.01, N02.FR.11 |
| System under test | Charging Station |
| Description | This test case describes how the CSMS requests the Charging Station to send a report about configured monitoring settings per component and variable. Optionally, this list can be filtered on monitoringCriteria and componentVariables. |
| Purpose | To verify if the Charging station is able to correctly report all monitoring data as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: The following monitors must be present as 'hard-wired' or 'preconfigured' or must have been configured by CSMS: - Component "ChargingStation", variable "AvailabilityState", monitor type <i>Delta</i> - Component "EVSE", Configured evse, variable "AvailabilityState", monitor type <i>Delta</i> |
| Reusable State: N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a GetMonitoringReportResponse | 1. The Test System sends a GetMonitoringReportRequest with monitoringCriteria omitted AND componentVariable omitted. |
| 3. The Charging Station sends a NotifyMonitoringReportRequest | 4. The Test System responds with a NotifyMonitoringReportResponse . |
| Note(s): - If tbc is <i>True</i> at Step 3 then step 3 and 4 will be repeated | |

| |
|---|
| Tool validations |
| * Step 3: Message: NotifyMonitoringReportRequest - requestId = <Generated requestId> While tbc = <i>true</i> , Message: NotifyMonitoringReportRequest - monitor.variable = "AvailabilityState" - monitor.variableMonitoring.type = <i>Delta</i> - monitor.component_name = <i>ChargingStation</i> or <i>EVSE</i> |
| Post scenario validations: - All reports have been received |

TC_N_48_CS: Alert Event - Variable monitoring on write only

| | |
|-------------------|---|
| Test case name | Alert Event - Variable monitoring on write only |
| Test case Id | TC_N_48_CS |
| Use case Id(s) | N07 |
| Requirement(s) | N07.FR.10 |
| System under test | Charging Station |
| Description | NotifyEventRequest reports every Component/Variable for which a VariableMonitoring setting was triggered. Only the VariableMonitoring settings that are responsible for triggering an event are included. |
| Purpose | To verify if the Charging station is able to correctly omit the actualField when a variablemonitor has been set to write only as described at the OCPP specification. |
| Prerequisite(s) | The Charging Station should be able to set a monitor on SecurityCtrlr.BasicAuthPassword and should be able to use security profile 1 or 2 |

| |
|--|
| Before (Preparations) |
| Configuration State: Security profile 1 or 2 is configured |
| Memory State: A Delta variableMonitoring setting has been set on a SecurityCtrlr.BasicAuthPassword |
| Reusable State: N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a SetVariablesResponse . | 1. The Test System sends a SetVariablesRequest with component.name = <i>SecurityCtrlr</i> variable.name = <i>BasicAuthPassword</i> attributeValue = <i><Generated password with same length as the configured basicAuthPassword></i> |
| 3. Execute Reusable State <i>Booted</i> . <u>Notes:</u> This step only needs to be executed when SetVariablesResponse status is <i>RebootRequired</i> . | |
| 4. The Charging station sends a NotifyEventRequest | 5. The Test System responds with a NotifyEventResponse . |

| |
|--|
| Tool validations |
| * Step 2: Message SetVariablesResponse - status must be <i>Accepted</i> or <i>RebootRequired</i> |
| * Step 4: Message NotifyEventRequest - eventData[0].actualValue must be an empty string |
| Post scenario validations: - N/a |

TC_N_61_CS: Alert Event - Variable monitoring on numeric

| | |
|-------------------|---|
| Test case name | Alert Event - Variable monitoring on numeric |
| Test case Id | TC_N_61_CS |
| Use case Id(s) | N07 |
| Requirement(s) | N07.FR.10 |
| System under test | Charging Station |
| Description | NotifyEventRequest reports every Component/Variable for which a VariableMonitoring setting was triggered. Only the VariableMonitoring settings that are responsible for triggering an event are included. |
| Purpose | To verify if the Charging station is able to correctly respond when a numeric Delta monitor is matched and exceeded, as described at the OCPP specification. |
| Prerequisite(s) | The Charging Station should be able to set a monitor on OCPPCommCtrlr.OfflineThreshold |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: A Delta variableMonitoring setting has been set on a OCPPCommCtrlr.OfflineThreshold |
| Reusable State: N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a SetVariablesResponse . | 1. The Test System sends a SetVariablesRequest with component.name = <i>OCPPCommCtrlr</i> variable.name = <i>OfflineThreshold</i> attributeValue = <i>Current Threshold + 1</i> |
| 3. Execute Reusable State <i>Booted</i> . <u>Notes:</u> This step only needs to be executed when SetVariablesResponse status is <i>RebootRequired</i> . | |
| <u>Notes:</u> The CS should not send a <i>NotifyEvent</i> as the delta monitor was not exceeded. | |
| 5. The Charging Station responds with a SetVariablesResponse . | 4. The Test System sends a SetVariablesRequest with component.name = <i>OCPPCommCtrlr</i> variable.name = <i>OfflineThreshold</i> attributeValue = <i>Current Threshold + 2</i> |
| 6. Execute Reusable State <i>Booted</i> . <u>Notes:</u> This step only needs to be executed when SetVariablesResponse status is <i>RebootRequired</i> . | |
| 7. The Charging station sends a NotifyEventRequest | 8. The Test System responds with a NotifyEventResponse . |

| |
|---|
| Tool validations |
| <p>* Step 2: Message SetVariablesResponse - status must be <i>Accepted</i> or <i>RebootRequired</i>+ * Step 5: Message SetVariablesResponse - status must be <i>Accepted</i> or <i>RebootRequired</i>+ * Step 7: Message NotifyEventRequest - eventData[0].actualValue must be <i>Current Threshold + 2</i></p> |
| Post scenario validations: - N/a |

TC_N_51_CS: Set Variable Monitoring - Modifying a VariableMonitor and trigger

| | |
|-------------------|---|
| Test case name | Set Variable Monitoring - Modifying a VariableMonitor and trigger |
| Test case Id | TC_N_51_CS |
| Use case Id(s) | N07 |
| Requirement(s) | N07.FR.11 |
| System under test | Charging Station |
| Description | NotifyEventRequest reports every Component/Variable for which a VariableMonitoring setting was triggered. Only the VariableMonitoring settings that are responsible for triggering an event are included. |
| Purpose | To verify if the Charging station is able to correctly check if the current value exceeds the new threshold as described at the OCPP specification. |
| Prerequisite(s) | Charging Station supports Monitoring |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: Variable monitor is already set with: setMonitoringData.component <Configured threshold monitor component variable> setMonitoringData.variable <Configured threshold monitor component variable> setMonitoringData.value <Configured threshold monitor component variable UpperThreshold non-trigger value> setMonitoringData.type UpperThreshold setMonitoringData.severity 5 Set MonitoringLevel to 8 <u>Notes:</u> If componentVariable is set to "Power" or "Current", the value is set to the configured maxLimit -1 |
| Reusable State: N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Notes:</u> If componentVariable is set to "Power" or "Current" EnergyTransferStarted will trigger the monitor. If another componentvariable is chosen a manual action is needed to trigger the monitor. | |
| 1. Execute Reusable State EnergyTransferStarted or manually trigger the monitor. | |
| 3. The Charging Station responds with a SetVariableMonitoringResponse | 2. The Test System sends a SetVariableMonitoringRequest with setMonitoringData.component <Configured threshold monitor component variable> setMonitoringData.variable <Configured threshold monitor component variable> setMonitoringData.id <SetVariableMonitoringResponse.setMonitoringResult.id in preparation phase> setMonitoringData.value <Configured threshold monitor component variable UpperThreshold trigger value> setMonitoringData.type UpperThreshold <u>Notes:</u> If componentVariable is set to "Power" or "Current", the value is set to 0.0 |
| 4. The Charging station sends a NotifyEventRequest | 5. The Test System responds with a NotifyEventResponse . |

| Tool validations |
|--|
| <div>* Step 3: Message SetVariableMonitoringResponse<ul style="list-style-type: none">- setMonitoringResult[0].status <i>Accepted</i>- setMonitoringResult[0].type <i>UpperThreshold</i>- setMonitoringResult[0].severity <i>5</i>- setMonitoringResult[0].component <i><Configured threshold monitor component variable></i>- setMonitoringResult[0].variable <i><Configured threshold monitor component variable></i></div> <div>* Step 4: Message NotifyEventRequest<ul style="list-style-type: none">- eventData[0].trigger <i>Alerting</i>- eventData[0].actualValue <i>> <Configured threshold monitor component variable UpperThreshold non-trigger value></i></div> |
| <div>Post scenario validations:<ul style="list-style-type: none">- All report parts have been received</div> |

TC_N_52_CS: Set Variable Monitoring - Removing a VariableMonitor

| | |
|-------------------|--|
| Test case name | Set Variable Monitoring - Removing a VariableMonitor |
| Test case Id | TC_N_52_CS |
| Use case Id(s) | N07 |
| Requirement(s) | N07.FR.12 |
| System under test | Charging Station |
| Description | NotifyEventRequest reports every Component/Variable for which a VariableMonitoring setting was triggered. Only the VariableMonitoring settings that are responsible for triggering an event are included. |
| Purpose | To verify if the Charging station is able to correctly communicate when a threshold has been exceeded and the applicable monitor is removed as described at the OCPP specification. |
| Prerequisite(s) | Charging Station supports Monitoring. If <i>Power</i> or <i>Current</i> is used as the monitored variable, then power must flow when a transaction reaches EnergyTransfer, or else the monitor (power or current > 0) is not triggered. |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: Variable monitor is already set with: setMonitoringData.component <Configured threshold monitor component variable> setMonitoringData.variable <Configured threshold monitor component variable> setMonitoringData.value <Configured threshold monitor component variable UpperThreshold trigger value> setMonitoringData.type UpperThreshold setMonitoringData.severity = 5 Set MonitoringLevel to 8 <u>Notes:</u> If componentVariable is set to "Power" or "Current", the value is set to 0.0 |
| Reusable State: N/a |

| Main (Test scenario) | |
|--|---|
| Charging Station | CSMS |
| 1. Execute Reusable State <i>EnergyTransferStarted</i> or manually trigger the monitor. <u>Notes:</u> If componentVariable is set to "Power" or "Current" EnergyTransferStarted will trigger the monitor. If another componentVariable is chosen a manual action is needed to trigger the monitor. | |
| 3. The Charging Station responds with a ClearVariableMonitoringResponse | 2. The Test System sends a ClearVariableMonitoringRequest with id <monitoringId of monitor set in Memory State> |
| 5. The Charging Station responds with a GetMonitoringReportResponse | 4. The Test System sends a GetMonitoringReportRequest with componentVariable.component <Configured threshold monitor component variable> componentVariable.variable <Configured threshold monitor component variable> monitoringCriteria ThresholdMonitoring |
| 6. Execute Reusable State <i>StopAuthorized</i> or manually trigger the monitor. <u>Notes:</u> If componentVariable is set to "Power" or "Current" EnergyTransferStarted will trigger the monitor. If another componentvariable is chosen a manual action is needed to trigger the monitor. <u>Note:</u> The Charging Station should not send a request for the cleared monitor | |

| Tool validations |
|---|
| <div>* Step 1: Message NotifyEventRequest - eventData[0].trigger <i>Alerting</i> - eventData[0].component <i><Configured threshold monitor component variable></i> - eventData[0].variable <i><Configured threshold monitor component variable></i> - eventData[0].variableMonitoringId <i><monitoringId of monitor set in Memory State></i></div> |
| <div>* Step 3: Message ClearVariableMonitoringResponse - clearMonitoringResult[0].status <i>Accepted</i> AND - clearMonitoringResult[0].id <i><monitoringId of monitor set in Memory State></i></div> |
| <div>* Step 5: Message GetMonitoringReportResponse - getMonitoringResult[0].status <i>EmptyResultSet</i></div> |
| <div>Post scenario validations: - No NotifyEventRequest that variableMontioringId <i><monitoringId of monitor set in Memory State></i> is cleared, is sent.</div> |

TC_N_53_CS: Alert Event - Persistent over reboot

| | |
|-------------------|---|
| Test case name | Alert Event - Persistent over reboot |
| Test case Id | TC_N_53_CS |
| Use case Id(s) | N07 |
| Requirement(s) | N07.FR.13 |
| System under test | Charging Station |
| Description | NotifyEventRequest reports every Component/Variable for which a VariableMonitoring setting was triggered. Only the VariableMonitoring settings that are responsible for triggering an event are included. |
| Purpose | To verify if the Charging station is able to save the variableMonitor data persistent across reboot as described at the OCPP specification. |
| Prerequisite(s) | n/a |

Before (Preparations)

Configuration State:

N/a

Memory State:

Variable monitor is already set with:

setMonitoringData.component <Configured threshold monitor component variable>**setMonitoringData.variable** <Configured threshold monitor component variable>**setMonitoringData.value** <Configured threshold monitor component variable UpperThreshold trigger value>**setMonitoringData.type** UpperThreshold

Reusable State:

Execute **Reusable State** *Booted*

Main (Test scenario)

| Charging Station | CSMS |
|--|--|
| 2. The Charging Station responds with a GetMonitoringReportResponse | 1. The Test System sends a GetMonitoringReportRequest with monitoringCriteria <i>ThresholdMonitoring</i> |
| 3. The Charging Station sends a NotifyMonitoringReportRequest | 4. The Test System responds with a NotifyMonitoringReportResponse . |

Note(s):

- If **tbc** is True at Step 3 then step 3 and 4 will be repeated

Tool validations

* Step 3:

Message **NotifyMonitoringReportRequest**- **requestId** <The Id of the request> AND- **monitor.variableMonitoring.id** <SetVariableMonitoringResponse.setMonitoringResult.id in preparation phase> -**monitor.variableMonitoring.type** UpperThreshold

Post scenario validations:

- All reports have been received

TC_N_56_CS: Alert Event - Delta value NOT numeric exceeded

| | |
|-------------------|---|
| Test case name | Alert Event - Delta value NOT numeric exceeded |
| Test case Id | TC_N_56_CS |
| Use case Id(s) | N07 |
| Requirement(s) | N07.FR.06, N07.FR.07, N07.FR.18, N07.FR.19 |
| System under test | Charging Station |
| Description | NotifyEventRequest reports every Component/Variable for which a VariableMonitoring setting was triggered. Only the VariableMonitoring settings that are responsible for triggering an event are included. |
| Purpose | To verify if the Charging station is correctly communicating when a delta value has exceeded as described at the OCPP specification. |
| Prerequisite(s) | n/a |

Before (Preparations)

Configuration State:

N/a

Memory State:

Variable monitor is configured with:

- **setMonitoringData[0].value** = 1
- **setMonitoringData[0].type** = *Delta*
- **setMonitoringData[0].severity** = 5
- **setMonitoringData[0].component** = <Configured non-numeric delta component variable>
- **setMonitoringData[0].variable** = <Configured non-numeric delta component variable>
- + Set MonitoringLevel to 8
- + Notes:
- Take a non-numeric component variable which can easily modified to trigger the alert.

Reusable State:

N/a

Main (Test scenario)

| | |
|---|---|
| Charging Station | CSMS |
| <u>Manual Action:</u> Make sure the configured delta value has been exceeded | |
| 1. The Charging Station sends a NotifyEventRequest | 2. The Test System responds with a NotifyEventResponse . |
| <u>Note(s):</u> - If tbc is True at Step 1 then step 1 and 2 will be repeated | |

Tool validations

* Step 1:

Message **NotifyEventRequest**

- **eventData[0].trigger** *Delta*
- **eventData[0].component** <Configured non-numeric delta component variable>
- **eventData[0].variable** <Configured non-numeric delta component variable>
- **eventData[0].variableMonitoringId** monitoringId of monitor set in Memory State

Post scenario validations:

- N/a

0 Display Message

TC_O_01_CS: Set Display Message - Success

| | |
|-------------------|---|
| Test case name | Set Display Message - Success |
| Test case Id | TC_O_01_CS |
| Use case Id(s) | O01 |
| Requirement(s) | O01_FR_12 |
| System under test | Charging Station |
| Description | This test case describes how the CSMS can be requested to sent an SetDisplayMessageRequest to the charging station. Depending on the given parameters the message shall be displayed a certain way and at a certain moment on the Charging Station. These messages are displayed additionally on a Charging Station and are not part of the firmware. |
| Purpose | To verify if the Charging Station is able to display additional messages according to the DisplayMessage mechanism as described in the OCPP specification. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a SetDisplayMessageResponse | 1. The Test System sends a SetDisplayMessageRequest with message.id <Generated displayMessageId> message.priority <Configured priority> message.display <Configured display component variable> (Omitted when left empty) |
| Note(s): - The display message is displayed as configured | |
| 4. The Charging Station responds with a GetDisplayMessagesResponse | 3. The Test System sends a GetDisplayMessagesRequest with id <Generated displayMessageId> requestId <Generated requestId> |
| 5. The Charging Station sends a NotifyDisplayMessagesRequest | 6. The Test System responds with a NotifyDisplayMessagesResponse . |

| |
|---|
| Tool validations |
| <p>* Step 2: Message SetDisplayMessageResponse - status Accepted</p> <p>* Step 4: Message GetDisplayMessagesResponse - status Accepted</p> <p>* Step 5: Message NotifyDisplayMessagesRequest - requestId <RequestId sent in step 3> - id <Generated id> - priority <Configured Priority> - message.format <Configured format> - message.content <Configured content></p> |

| |
|--|
| Tool validations |
| Post scenario validations: - N/a |

TC_O_02_CS: Get all Display Messages - Success

| | |
|-------------------|--|
| Test case name | Get all Display Messages - Success |
| Test case Id | TC_O_02_CS |
| Use case Id(s) | O03 |
| Requirement(s) | O03_FR_01, O03_FR_02, O03_FR_03, O03_FR_04, O03_FR_05 |
| System under test | Charging Station |
| Description | This test case describes how a CSO can request all the installed DisplayMessages configured via OCPP in a Charging Station. The Charging Station can remove messages when they are out-dated, or transactions have ended. It can be very useful for a CSO to be able to view the current list of messages, so the CSO knows which messages are (still) configured. |
| Purpose | To verify if the Charging Station is able to send the requested DisplayMessages according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | N/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: SetDisplayMessage |
| Reusable State: N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a GetDisplayMessagesResponse | 1. The Test System sends a GetDisplayMessagesRequest requestId <Generated requestId> |
| 3. The Charging Station sends a NotifyDisplayMessagesRequest | 4. The Test System responds with a NotifyDisplayMessagesResponse . |
| <u>Note(s):</u> - If tbc is True at Step 3 then step 3 and 4 will be repeated | |

| |
|--|
| Tool validations |
| * Step 2: Message GetDisplayMessagesResponse - status Accepted * Step 3: Message NotifyDisplayMessagesRequest - requestId <Generated requestId> |
| Post scenario validations: - All messages have been received |

TC_O_03_CS: Get all Display Messages - No DisplayMessages configured

| | |
|--------------------------|---|
| Test case name | Get all Display Messages - No DisplayMessages configured |
| Test case Id | TC_O_03_CS |
| Use case Id(s) | O03 |
| Requirement(s) | O03_FR_06 |
| System under test | Charging Station |
| Description | This test case describes how a CSO can request all the installed DisplayMessages configured via OCPP in a Charging Station. The Charging Station can remove messages when they are out-dated, or transactions have ended. It can be very useful for a CSO to be able to view to current list of messages, so the CSO knows which messages are (still) configured. |
| Purpose | To verify if the Charging Station is responding according to the DisplayMessage mechanism as described in the OCPP specification when no Display Messages are configured. |
| Prerequisite(s) | N/a |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a GetDisplayMessagesResponse | 1. The Test System sends a GetDisplayMessagesRequest |

| |
|--|
| Tool validations |
| * Step 2: Message GetDisplayMessagesResponse - status <i>Unknown</i> |
| Post scenario validations: - N/a |

TC_O_04_CS: Clear Display Message - Success

| | |
|-------------------|---|
| Test case name | Clear Display Message - Success |
| Test case Id | TC_O_04_CS |
| Use case Id(s) | O05 |
| Requirement(s) | O05_FR_01 |
| System under test | Charging Station |
| Description | This test case describes how a CSO can remove a specific message, configured via OCPP in a Charging Station. |
| Purpose | To verify if the Charging Station is able to remove a specific message requested by the CSMS according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | N/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: SetDisplayMessage |
| Reusable State: N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a ClearDisplayMessageResponse | 1. The Test System sends a ClearDisplayMessageRequest with id <Generated displayMessageId> |
| 4. The Charging Station responds with a GetDisplayMessagesResponse | 3. The Test System sends a GetDisplayMessagesRequest with id <Generated displayMessageId> |

| |
|--|
| Tool validations |
| <p>* Step 2: Message ClearDisplayMessageResponse - status <i>Accepted</i></p> <p>* Step 4: Message: GetDisplayMessagesResponse - status must be <i>Unknown</i></p> |
| Post scenario validations: - N/a |

TC_O_05_CS: Clear Display Message - Unknown Key

| | |
|--------------------------|--|
| Test case name | Clear Display Message - Unknown Key |
| Test case Id | TC_O_05_CS |
| Use case Id(s) | O05 |
| Requirement(s) | O05_FR_02 |
| System under test | Charging Station |
| Description | This test case describes how a CSO can remove a specific message, configured via OCPP in a Charging Station. |
| Purpose | To verify if the Charging Station is able to respond according the mechanism as described in the OCPP specification when no message is configured with the specified id. |
| Prerequisite(s) | N/a |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a ClearDisplayMessageResponse | 1. The Test System sends a ClearDisplayMessageRequest with id <Generated displayMessageId> |

| |
|---|
| Tool validations |
| * Step 2: Message ClearDisplayMessageResponse - status <i>Unknown</i> |
| Post scenario validations: - N/a |

TC_O_06_CS: Set Display Message - Specific transaction - Success

| | |
|-------------------|--|
| Test case name | Set Display Message - Specific transaction - Success |
| Test case Id | TC_O_06_CS |
| Use case Id(s) | 002 |
| Requirement(s) | 002.FR.02, 002_FR_14 |
| System under test | Charging Station |
| Description | This test case describes how a CSO can set a message to be displayed on a Charging Station for a specific transaction. Depending on the given parameters the message shall be displayed a certain way on the Charging Station. |
| Purpose | To verify if the Charging Station is able to display the message correctly according the mechanism as described in the OCPP specification when a transaction is ongoing. |
| Prerequisite(s) | N/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: State is <i>EnergyTransferStarted</i> |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a SetDisplayMessageResponse | 1. The Test System sends a SetDisplayMessageRequest with message.id <Generated displayMessageId> AND message.transactionId <Configured transactionId> AND message.priority <Configured Priority AND message.display <Configured display component variable> (Omitted when left empty) |
| Note(s): - The display message is displayed as configured | |
| 3. Execute Reusable State <i>StopAuthorized</i> | |
| 4. Execute Reusable State <i>EVConnectedPostSession</i> | |
| 5. Execute Reusable State <i>EVDisconnected</i> | |
| 6. Execute Reusable State <i>ParkingBayUnoccupied</i> | |
| Note(s): - The display message is not displayed anymore | |
| 8. The Charging Station responds with a GetDisplayMessagesResponse | 7. The Test System sends a GetDisplayMessagesRequest with id <Generated displayMessageId> |

| |
|---|
| Tool validations |
| * Step 1: Message: SetDisplayMessageResponse - status must be <i>Accepted</i> |
| * Step 8: Message: GetDisplayMessagesResponse - status must be <i>Unknown</i> |
| Post scenario validations: N/a |

TC_O_07_CS: Get a Specific Display Message - Id

| | |
|-------------------|--|
| Test case name | Get a Specific Display Message - Id |
| Test case Id | TC_O_07_CS |
| Use case Id(s) | 004 |
| Requirement(s) | 004_FR_01, 004_FR_03, 004_FR_04, 004_FR_05, 004_FR_06 |
| System under test | Charging Station |
| Description | This test case describes how a CSO can request specific installed DisplayMessages configured via OCPP in a Charging Station. The Charging Station can remove messages when they are out-dated, or transactions have ended. It can be very useful for a CSO to be able to view to current list of messages, so the CSO knows which messages are (still) configured. |
| Purpose | To verify if the Chargin Station is able to respond the specific id message requested by the CSMS according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | N/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: SetDisplayMessage |
| Reusable State: N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a GetDisplayMessagesResponse | 1. The Test System sends a GetDisplayMessagesRequest with id <Generated displayMessageId> requestId <Generated requestId> |
| 3. The Charging Station sends a NotifyDisplayMessagesRequest | 4. The Test System responds with a NotifyDisplayMessagesResponse . |
| <u>Note(s):</u> - If tbc is True at Step 3 then step 3 and 4 will be repeated | |

| |
|---|
| Tool validations |
| <p>* Step 2:</p> <p>Message GetDisplayMessagesResponse</p> <p>- status Accepted</p> <p>* Step 3:</p> <p>Message NotifyDisplayMessagesRequest</p> <p>- requestId <Generated requestId></p> |
| <p>Post scenario validations:</p> <p>- All messages have been received</p> |

TC_O_08_CS: Get a Specific Display Message - Priority

| | |
|-------------------|--|
| Test case name | Get a Specific Display Message - Priority |
| Test case Id | TC_O_08_CS |
| Use case Id(s) | 004 |
| Requirement(s) | 004_FR_01, 004_FR_03, 004_FR_04, 004_FR_05, 004_FR_06 |
| System under test | Charging Station |
| Description | This test case describes how a CSO can request specific installed DisplayMessages configured via OCPP in a Charging Station. The Charging Station can remove messages when they are out-dated, or transactions have ended. It can be very useful for a CSO to be able to view to current list of messages, so the CSO knows which messages are (still) configured. |
| Purpose | To verify if the Chargin Station is able to respond the specific priority messages requested by the CSMS according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | N/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: SetDisplayMessage |
| Reusable State: N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a GetDisplayMessagesResponse | 1. The Test System sends a GetDisplayMessagesRequest with priority <Configured display_message_priority> requestId <Generated requestId> |
| 3. The Charging Station sends a NotifyDisplayMessagesRequest | 4. The Test System responds with a NotifyDisplayMessagesResponse . |
| <u>Note(s):</u> - If tbc is True at Step 3 then step 3 and 4 will be repeated | |

| |
|--|
| Tool validations |
| * Step 2: Message GetDisplayMessagesResponse - status Accepted * Step 3: Message NotifyDisplayMessagesRequest - requestId <Generated requestId> |
| Post scenario validations: - All messages have been received |

TC_O_09_CS: Get a Specific Display Message - State

| | |
|-------------------|--|
| Test case name | Get a Specific Display Message - State |
| Test case Id | TC_O_09_CS |
| Use case Id(s) | O04 |
| Requirement(s) | O04_FR_01, O04_FR_03, O04_FR_04, O04_FR_05, O04_FR_06 |
| System under test | Charging Station |
| Description | This test case describes how a CSO can request specific installed DisplayMessages configured via OCPP in a Charging Station. The Charging Station can remove messages when they are out-dated, or transactions have ended. It can be very useful for a CSO to be able to view to current list of messages, so the CSO knows which messages are (still) configured. |
| Purpose | To verify if the Chargin Station is able to respond the specific state messages requested by the CSMS according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | N/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: SetDisplayMessage with state <Configured Message state> |
| Reusable State: N/a |

| Main (Test scenario) | |
|---|---|
| Charging Station | CSMS |
| 2. The Charging Station responds with a GetDisplayMessagesResponse | 1. The Test System sends a GetDisplayMessagesRequest with state <Configured display_message_state> requestId <Generated requestId> |
| 3. The Charging Station sends a NotifyDisplayMessagesRequest | 4. The Test System responds with a NotifyDisplayMessagesResponse . |
| <u>Note(s):</u> - If tbc is True at Step 3 then step 3 and 4 will be repeated | |

| |
|--|
| Tool validations |
| * Step 2: Message GetDisplayMessagesResponse - status Accepted * Step 3: Message NotifyDisplayMessagesRequest - requestId <Generated requestId> |
| Post scenario validations: - All messages have been received |

TC_O_10_CS: Set Display Message - Specific transaction - UnknownTransaction

| | |
|-------------------|--|
| Test case name | Set Display Message - Specific transaction - UnknownTransaction |
| Test case Id | TC_O_10_CS |
| Use case Id(s) | 002 |
| Requirement(s) | 002_FR_01 |
| System under test | Charging Station |
| Description | This test case describes how a CSO can set a message to be displayed on a Charging Station for a specific transaction. Depending on the given parameters the message shall be displayed a certain way on the Charging Station. |
| Purpose | To verify if the Charging Station responds correctly according the mechanism as described in the OCPP specification when a display message request is received for an unknown specific transaction. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| Main (Test scenario) | |
|--|---|
| Charging Station | CSMS |
| 2. The Charging Station responds with a SetDisplayMessageResponse | 1. The Test System sends a SetDisplayMessageRequest with message.id <Generated displayMessageId> AND message.transactionId <Generated transactionId> AND message.priority <Configured Priority AND message.display <Configured display component variable> (Omitted when left empty) |

| |
|--|
| Tool validations |
| * Step 2: Message SetDisplayMessageResponse - status <i>UnknownTransaction</i> |
| Post scenario validations: - N/a |

TC_O_11_CS: Get a Specific Display Message - Unknown parameters

| | |
|-------------------|--|
| Test case name | Get a Specific Display Message - Unknown parameters |
| Test case Id | TC_O_11_CS |
| Use case Id(s) | 004 |
| Requirement(s) | 004_FR_02 |
| System under test | Charging Station |
| Description | This test case describes how a CSO can request specific installed DisplayMessages configured via OCPP in a Charging Station. The Charging Station can remove messages when they are out-dated, or transactions have ended. It can be very useful for a CSO to be able to view to current list of messages, so the CSO knows which messages are (still) configured. |
| Purpose | To verify if the Chargin Station is able to respond correctly according to the mechanism as described in the OCPP specification when the specific id message requested by the CSMS is unknown. |
| Prerequisite(s) | N/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: SetDisplayMessage |
| Reusable State: N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a GetDisplayMessagesResponse | 1. The Test System sends a GetDisplayMessagesRequest with id <Other generated messageId> |

| |
|--|
| Tool validations |
| * Step 2: Message GetDisplayMessagesResponse - status <i>Unknown</i> |
| Post scenario validations: - N/a |

TC_O_12_CS: Set Display Message - Replace DisplayMessage

| | |
|-------------------|---|
| Test case name | Set Display Message - Replace DisplayMessage |
| Test case Id | TC_O_12_CS |
| Use case Id(s) | 006 |
| Requirement(s) | 006_FR_01 |
| System under test | Charging Station |
| Description | This test case describes how a CSO can replace a DisplayMessage that is previously configured in a Charging Station. Replace the message content, but also all the given parameters with the new one. |
| Purpose | To verify if the Chargin Station is able to replace a display message according to the DisplayMessage mechanism as described in the OCPP specification. |
| Prerequisite(s) | N/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: SetDisplayMessage |
| Reusable State: N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a SetDisplayMessageResponse | 1. The Test System sends a SetDisplayMessageRequest with message.id <Generated displayMessageId from before (preperation) steps> message.message.content <Different message to indicate the message was replaced> |
| <u>Note(s):</u> - The display message is replaced by a new one. | |

| |
|--|
| Tool validations |
| * Step 2: Message SetDisplayMessageResponse - status <i>Accepted</i> |
| Post scenario validations: - N/a |

TC_O_13_CS: Set Display Message - Display message at StartTime

| | |
|-------------------|---|
| Test case name | Set Display Message - Display message at StartTime |
| Test case Id | TC_O_13_CS |
| Use case Id(s) | 001 |
| Requirement(s) | 001_FR_06 |
| System under test | Charging Station |
| Description | This test case describes how the CSMS can be requested to sent an SetDisplayMessageRequest to the charging station. Depending on the given parameters the message shall be displayed a certain way and at a certain moment on the Charging Station. These messages are displayed additionally on a Charging Station and are not part of the firmware. |
| Purpose | To verify if the Charging Station is able to display additional messages with a certain start time according to the DisplayMessage mechanism as described in the OCPP specification. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a SetDisplayMessageResponse | 1. The Test System sends a SetDisplayMessageRequest with message.id <Generated displayMessageId> message.priority <Configured Priority> message.startDateTime <Current dateTime + Configured Start Date Time Offset> message.display <Configured display component variable> (Omitted when left empty) |
| 4. The Charging Station responds with a GetDisplayMessagesResponse | 3. The Test System sends a GetDisplayMessagesRequest with id <Generated displayMessageId> |
| 5. The Charging Station sends a NotifyDisplayMessagesRequest | 6. The Test System responds with a NotifyDisplayMessagesResponse . |

Note(s):

- If **ttbc** is True at Step 5 then step 5 and 6 will be repeated
- Wait till <Configured Start Date Time Offset> seconds have passed
- The display message should be displayed after <Configured Start Date Time Offset> seconds.

| |
|---|
| Tool validations |
| <p>* Step 2: Message SetDisplayMessageResponse - status Accepted</p> <p>* Step 4: Message GetDisplayMessagesResponse - status Accepted</p> <p>* Step 5: Message NotifyDisplayMessagesRequest - requestId <Generated requestId> - startDateTime <Should not be Omitted.></p> |

| |
|--|
| Tool validations |
| Post scenario validations: - N/a |

TC_O_14_CS: Set Display Message - Remove message after EndTime

| | |
|-------------------|---|
| Test case name | Set Display Message - Remove message after EndTime |
| Test case Id | TC_O_14_CS |
| Use case Id(s) | 001 |
| Requirement(s) | 001_FR_07 |
| System under test | Charging Station |
| Description | This test case describes how the CSMS can be requested to sent an SetDisplayMessageRequest to the charging station. Depending on the given parameters the message shall be displayed a certain way and at a certain moment on the Charging Station. These messages are displayed additionally on a Charging Station and are not part of the firmware. |
| Purpose | To verify if the Charging Station is able to display additional messages with a certain end time according to the DisplayMessage mechanism as described in the OCPP specification. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a SetDisplayMessageResponse | 1. The Test System sends a SetDisplayMessageRequest with message.id <Generated displayMessageId> message.priority <Configured Priority> message.endDateTime <Current dateTime + Configured End Date Time Offset> message.display <Configured display component variable> (Omitted when left empty) |
| 4. The Charging Station responds with a GetDisplayMessagesResponse | 3. The Test System sends a GetDisplayMessagesRequest with id <Generated displayMessageId> |
| 5. The Charging Station sends a NotifyDisplayMessagesRequest | 6. The Test System responds with a NotifyDisplayMessagesResponse . |
| <u>Note(s):</u> - If tbc is True at Step 5 then step 5 and 6 will be repeated - Wait till <Configured End Date Time Offset> seconds have passed - The display message is displayed and removed after <Configured End Date Time Offset> seconds. | |
| 8. The Charging Station responds with a GetDisplayMessagesResponse | 7. The Test System sends a GetDisplayMessagesRequest with id <Generated displayMessageId> requestId <Generated requestId> |

| Tool validations |
|--|
| <div>* Step 2: Message SetDisplayMessageResponse - status <i>Accepted</i></div> <div>* Step 4: Message GetDisplayMessagesResponse - status <i>Accepted</i></div> <div>* Step 5: Message NotifyDisplayMessagesRequest - requestId <i><Generated requestId></i> - endDateTime <i><Should not be Omitted.></i></div> <div>* Step 8: Message GetDisplayMessagesResponse - status <i>Unknown</i></div> |
| <div>Post scenario validations: - N/a</div> |

TC_O_17_CS: Set Display Message - NotSupportedPriority

| | |
|-------------------|---|
| Test case name | Set Display Message - NotSupportedPriority |
| Test case Id | TC_O_17_CS |
| Use case Id(s) | 001 |
| Requirement(s) | 001.FR.01, 002.FR.03 |
| System under test | Charging Station |
| Description | This test case describes how the CSMS can be requested to sent an SetDisplayMessageRequest to the charging station. Depending on the given parameters the message shall be displayed a certain way and at a certain moment on the Charging Station. These messages are displayed additionally on a Charging Station and are not part of the firmware. |
| Purpose | To verify if the Charging Station is able to respond correctly when the priority of the display messages is not supported according to the DisplayMessage mechanism as described in the OCPP specification. |
| Prerequisite(s) | Charging station should not support all priorities described in the OCPP specification |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| Main (Test scenario) | |
|--|---|
| Charging Station | CSMS |
| 2. The Charging Station responds with a SetDisplayMessageResponse | 1. The Test System sends a SetDisplayMessageRequest with message.id <Generated displayMessageId> message.priority <Configured unsupported_display_message_priority> message.display <Configured display component variable> (Omitted when left empty) |

| |
|--|
| Tool validations |
| * Step 2: Message SetDisplayMessageResponse - status NotSupportedPriority |
| Post scenario validations: - N/a |

TC_O_18_CS: Set Display Message - NotSupportedState

| | |
|-------------------|---|
| Test case name | Set Display Message - NotSupportedState |
| Test case Id | TC_O_18_CS |
| Use case Id(s) | 001 |
| Requirement(s) | 001_FR_02, 002.FR.04 |
| System under test | Charging Station |
| Description | This test case describes how the CSMS can be requested to sent an SetDisplayMessageRequest to the charging station. Depending on the given parameters the message shall be displayed a certain way and at a certain moment on the Charging Station. These messages are displayed additionally on a Charging Station and are not part of the firmware. |
| Purpose | To verify if the Charging Station is able to respond correctly when the state of the display messages is not supported according to the DisplayMessage mechanism as described in the OCPP specification. |
| Prerequisite(s) | Charging station should not support all states described in the OCPP specification |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State:

N/a

Main (Test scenario)

| Charging Station | CSMS |
|---|--|
| 2. The Charging Station responds with a SetDisplayMessageResponse | 1. The Test System sends a SetDisplayMessageRequest with message.id <Generated displayMessageId> message.state <Configured unsupported_display_message_state> message.display <Configured display component variable> (Omitted when left empty) |

Tool validations

* Step 2:

Message SetDisplayMessageResponse

- **status** NotSupportedState

Post scenario validations:

- N/a

TC_O_19_CS: Set Display Message - NotSupportedMessageFormat

| | |
|-------------------|---|
| Test case name | Set Display Message - NotSupportedMessageFormat |
| Test case Id | TC_O_19_CS |
| Use case Id(s) | 001 |
| Requirement(s) | 001_FR_03, 002.FR.05 |
| System under test | Charging Station |
| Description | This test case describes how the CSMS can be requested to sent an SetDisplayMessageRequest to the charging station. Depending on the given parameters the message shall be displayed a certain way and at a certain moment on the Charging Station. These messages are displayed additionally on a Charging Station and are not part of the firmware. |
| Purpose | To verify if the Charging Station is able to respond correctly when the message format of the display messages is not supported according to the DisplayMessage mechanism as described in the OCPP specification. |
| Prerequisite(s) | The Charging station does not support all formats described in the OCPP specification |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State:

N/a

Main (Test scenario)

| Charging Station | CSMS |
|---|--|
| 2. The Charging Station responds with a SetDisplayMessageResponse | 1. The Test System sends a SetDisplayMessageRequest with message.id <Generated displayMessageId> message.message.format <Configured Unsupported Message Format> message.display <Configured display component variable> (Omitted when left empty) |

Tool validations

* Step 2:

Message SetDisplayMessageResponse

- **status** NotSupportedMessageFormat

Post scenario validations:

- N/a

TC_O_20_CS: Set Display Message - Persistent over reboot

| | |
|-------------------|---|
| Test case name | Set Display Message - Persistent over reboot |
| Test case Id | TC_O_20_CS |
| Use case Id(s) | 001 |
| Requirement(s) | 001_FR_10 |
| System under test | Charging Station |
| Description | This test case describes how the CSMS can be requested to sent an SetDisplayMessageRequest to the charging station. Depending on the given parameters the message shall be displayed a certain way and at a certain moment on the Charging Station. These messages are displayed additionally on a Charging Station and are not part of the firmware. |
| Purpose | To verify if the Charging Station is able to store display messages persistent over reboot according to the DisplayMessage mechanism as described in the OCPP specification. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a SetDisplayMessageResponse | 1. The Test System sends a SetDisplayMessageRequest with message.id <Generated displayMessageId> message.priority <Configured Priority> message.display <Configured display component variable> (Omitted when left empty) |
| 3. Execute Reusable State <i>Booted</i> | |
| 5. The Charging Station responds with a GetDisplayMessagesResponse | 4. The Test System sends a GetDisplayMessagesRequest with id <Generated displayMessageId> requestId <Generated requestId> |
| 6. The Charging Station sends a NotifyDisplayMessagesRequest | 7. The Test System responds with a NotifyDisplayMessagesResponse . |
| Note(s): - If tlc is True at Step 5 then step 5 and 6 will be repeated | |

| |
|---|
| Tool validations |
| <p>* Step 2: Message SetDisplayMessageResponse - status Accepted</p> <p>* Step 5: Message GetDisplayMessagesResponse - status Accepted</p> <p>* Step 6: Message NotifyDisplayMessagesRequest - requestId <RequestId sent in step 4> - id <Generated id> - priority <Configured Priority> - message.format <Configured format> - message.content <Configured content></p> |

| |
|--|
| Tool validations |
| Post scenario validations: - N/a |

TC_O_22_CS: Set Display Message - Multiple In front priority

| | |
|-------------------|---|
| Test case name | Set Display Message - Multiple In front priority |
| Test case Id | TC_O_22_CS |
| Use case Id(s) | 001 |
| Requirement(s) | 001_FR_14 |
| System under test | Charging Station |
| Description | This test case describes how the CSMS can be requested to sent an SetDisplayMessageRequest to the charging station. Depending on the given parameters the message shall be displayed a certain way and at a certain moment on the Charging Station. These messages are displayed additionally on a Charging Station and are not part of the firmware. |
| Purpose | To verify if the Charging Station is able to display multiple additional messages with a "InFront" priority according to the DisplayMessage mechanism as described in the OCPP specification. |
| Prerequisite(s) | N/a |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State:

N/a

Main (Test scenario)

| Charging Station | CSMS |
|---|--|
| 2. The Charging Station responds with a SetDisplayMessageResponse | 1. The Test System sends a SetDisplayMessageRequest with message.id <Generated displayMessageId> message.priority InFront message.display <Configured display component variable> (Omitted when left empty) |
| 4. The Charging Station responds with a SetDisplayMessageResponse | 3. The Test System sends a SetDisplayMessageRequest with message.id <Generated displayMessage2Id> message.priority InFront message.display <Configured display component variable> (Omitted when left empty) |
| 6. The Charging Station responds with a GetDisplayMessagesResponse | 5. The Test System sends a GetDisplayMessagesRequest with id <Generated displayMessageId> requestId <Generated requestId> |
| 7. The Charging Station sends a NotifyDisplayMessagesRequest | 8. The Test System responds with a NotifyDisplayMessagesResponse . |
| 10. The Charging Station responds with a GetDisplayMessagesResponse | 9. The Test System sends a GetDisplayMessagesRequest with id <Generated displayMessage2Id> requestId <Generated requestId> |
| 11. The Charging Station sends a NotifyDisplayMessagesRequest | 12. The Test System responds with a NotifyDisplayMessagesResponse . |
| Note(s): - If tbc is True at Step 7 then step 7 and 8 will be repeated - If tbc is True at Step 11 then step 11 and 12 will be repeated - The display messages are displayed as configured according the priority | |

| Tool validations |
|---|
| <div>* Step 2: Message SetDisplayMessageResponse - status <i>Accepted</i></div> <div>* Step 4: Message SetDisplayMessageResponse - status <i>Accepted</i></div> <div>* Step 6: Message GetDisplayMessagesResponse - status <i>Accepted</i></div> <div>* Step 7: Message NotifyDisplayMessagesRequest - requestId <i><Generated requestId></i></div> <div>* Step 10: Message GetDisplayMessagesResponse - status <i>Accepted</i></div> <div>* Step 11: Message NotifyDisplayMessagesRequest - requestId <i><Generated requestId></i></div> |
| <div>Post scenario validations: - N/a</div> |

TC_O_24_CS: Set Display Message - Second Alwaysfront priority

| | |
|-------------------|---|
| Test case name | Set Display Message - Second Alwaysfront priority |
| Test case Id | TC_O_24_CS |
| Use case Id(s) | 001 |
| Requirement(s) | 001_FR_16 |
| System under test | Charging Station |
| Description | This test case describes how the CSMS can be requested to sent an SetDisplayMessageRequest to the charging station. Depending on the given parameters the message shall be displayed a certain way and at a certain moment on the Charging Station. These messages are displayed additionally on a Charging Station and are not part of the firmware. |
| Purpose | To verify if the Charging Station is able to display multiple additional messages with a "AlwaysFront" priority according to the DisplayMessage mechanism as described in the OCPP specification. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a SetDisplayMessageResponse | 1. The Test System sends a SetDisplayMessageRequest with message.id <Generated displayMessageId> message.priority AlwaysFront |
| 4. The Charging Station responds with a SetDisplayMessageResponse | 3. The Test System sends a SetDisplayMessageRequest with message.id <Configured displayMessage2Id> message.priority AlwaysFront |
| 6. The Charging Station responds with a GetDisplayMessagesResponse | 5. The Test System sends a GetDisplayMessagesRequest with id <Configured displayMessage2Id> |
| 7. The Charging Station sends a NotifyDisplayMessagesRequest | 8. The Test System responds with a NotifyDisplayMessagesResponse . |
| Note(s): - If tbc is True at Step 7 then step 7 and 8 will be repeated - The message from step 1 is NOT displayed anymore and is replaced by the message from step 5. | |

| |
|--|
| Tool validations |
| * Step 2: Message SetDisplayMessageResponse - status Accepted * Step 4: Message SetDisplayMessageResponse - status Accepted * Step 6: Message GetDisplayMessagesResponse - status Accepted * Step 7: Message NotifyDisplayMessagesRequest - requestId <Generated requestId> |

| |
|--|
| Tool validations |
| Post scenario validations: - N/a |

TC_O_27_CS: Set Display Message - Specific transaction - Display message at StartTime

| | |
|-------------------|--|
| Test case name | Set Display Message - Specific transaction - Display message at StartTime |
| Test case Id | TC_O_27_CS |
| Use case Id(s) | O02 |
| Requirement(s) | O02_FR_06 |
| System under test | Charging Station |
| Description | This test case describes how a CSO can set a message to be displayed on a Charging Station for a specific transaction. Depending on the given parameters the message shall be displayed a certain way on the Charging Station. |
| Purpose | To verify if the Charging Station is able to display the message with a certain start time correctly according the mechanism as described in the OCPP specification when a transaction is ongoing. |
| Prerequisite(s) | N/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: State is <i>EnergyTransferStarted</i> |

| Main (Test scenario) | |
|---|--|
| Charging Station | CSMS |
| 2. The Charging Station responds with a SetDisplayMessageResponse | 1. The Test System sends a SetDisplayMessageRequest with message.id <Generated displayMessageId> message.priority <Configured Priority> message.startDateTime <Current dateTime + Configured Start Date Time Offset> message.display <Configured display component variable> (Omitted when left empty) |
| <u>Note(s):</u> - The display message is not yet displayed. - Waiting <Configured Start Date Time Offset> seconds. - The display message is displayed after <Configured Start Date Time Offset> seconds. | |
| 3. Execute Reusable State <i>StopAuthorized</i> | |
| 4. Execute Reusable State <i>EVConnectedPostSession</i> | |
| 5. Execute Reusable State <i>EVDisconnected</i> | |
| 6. Execute Reusable State <i>ParkingBayUnoccupied</i> | |
| <u>Note(s):</u> - The display message is not displayed anymore | |
| 8. The Charging Station responds with a GetDisplayMessagesResponse | 7. The Test System sends a GetDisplayMessagesRequest with id <Generated displayMessageId> |

| |
|---|
| Tool validations |
| * Step 2: Message SetDisplayMessageResponse - status <i>Accepted</i> * Step 8: Message: GetDisplayMessagesResponse - status <i>Unknown</i> |

| |
|--|
| Tool validations |
| Post scenario validations: - N/a |

TC_O_28_CS: Set Display Message - Specific transaction - Remove message after EndTime

| | |
|-------------------|---|
| Test case name | Set Display Message - Specific transaction - Remove message after EndTime |
| Test case Id | TC_O_28_CS |
| Use case Id(s) | O02 |
| Requirement(s) | O02_FR_07 |
| System under test | Charging Station |
| Description | This test case describes how the CSMS can be requested to sent an SetDisplayMessageRequest to the charging station. Depending on the given parameters the message shall be displayed a certain way and at a certain moment on the Charging Station. These messages are displayed additionally on a Charging Station and are not part of the firmware. |
| Purpose | To verify if the Charging Station is able to display additional messages with a certain end time for a specific transaction according to the DisplayMessage mechanism as described in the OCPP specification. |
| Prerequisite(s) | N/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: State is <i>EnergyTransferStarted</i> |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a SetDisplayMessageResponse | 1. The Test System sends a SetDisplayMessageRequest with message.id <Generated displayMessageId> message.transactionId <Generated transactionId> message.priority <Configured Priority> message.endDateTime <Current dateTime + Configured End Date Time Offset> message.display <Configured display component variable> (Omitted when left empty) |
| <u>Note(s):</u> - The display message should be displayed. - Waiting <Configured End Date Time Offset> seconds. - The display message is not being displayed anymore after <Configured End Date Time Offset> seconds. | |
| 4. The Charging Station responds with a GetDisplayMessagesResponse | 3. The Test System sends a GetDisplayMessagesRequest with id <Generated displayMessageId> |

| |
|--|
| Tool validations |
| * Step 2: Message SetDisplayMessageResponse - status <i>Accepted</i> * Step 4: Message GetDisplayMessagesResponse - status <i>Unknown</i> |
| Post scenario validations: - N/a |

TC_O_30_CS: Set Display Message - Specific transaction - Multiple In front priority

| | |
|-------------------|--|
| Test case name | Set Display Message - Specific transaction - Multiple In front priority |
| Test case Id | TC_O_30_CS |
| Use case Id(s) | O02 |
| Requirement(s) | O02_FR_16 |
| System under test | Charging Station |
| Description | This test case describes how a CSO can set a message to be displayed on a Charging Station for a specific transaction. Depending on the given parameters the message shall be displayed a certain way on the Charging Station. |
| Purpose | To verify if the Charging Station is able to display multiple additional messages with a "InFront" priority for a specific transaction according to the DisplayMessage mechanism as described in the OCPP specification. |
| Prerequisite(s) | N/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: State is <i>EnergyTransferStarted</i> |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a SetDisplayMessageResponse | 1. The Test System sends a SetDisplayMessageRequest with message.id <Generated displayMessageId> AND message.transactionId <Received transactionId> AND message.priority <i>InFront</i> AND message.display <Configured display component variable> (Omitted when left empty) |
| 4. The Charging Station responds with a SetDisplayMessageResponse | 3. The Test System sends a SetDisplayMessageRequest with message.id <Generated displayMessageId2> AND message.transactionId <Received transactionId> AND message.priority <i>InFront</i> AND message.display <Configured display component variable> (Omitted when left empty) |
| Note(s): - The display messages are displayed as configured | |
| 6. The Charging Station responds with a GetDisplayMessagesResponse | 5. The Test System sends a GetDisplayMessagesRequest with id <Generated displayMessageId> requestId <Generated requestId> |
| 7. The Charging Station sends a NotifyDisplayMessagesRequest | 8. The Test System responds with a NotifyDisplayMessagesResponse . |
| 10. The Charging Station responds with a GetDisplayMessagesResponse | 9. The Test System sends a GetDisplayMessagesRequest with id <Generated displayMessageId2> requestId <Generated requestId> |
| 11. The Charging Station sends a NotifyDisplayMessagesRequest | 12. The Test System responds with a NotifyDisplayMessagesResponse . |
| 13. Execute Reusable State <i>StopAuthorized</i> | |
| 14. Execute Reusable State <i>EVConnectedPostSession</i> | |
| 15. Execute Reusable State <i>EVDisconnected</i> | |
| 16. Execute Reusable State <i>ParkingBayUnoccupied</i> | |

| Main (Test scenario) | |
|--|--|
| <u>Note(s):</u> - The display messages are not displayed anymore | |
| 18. The Charging Station responds with a GetDisplayMessagesResponse | 17. The Test System sends a GetDisplayMessagesRequest with id <Generated displayMessageId> |
| 20. The Charging Station responds with a GetDisplayMessagesResponse | 19. The Test System sends a GetDisplayMessagesRequest with id <Configured displayMessage2Id> |

| Tool validations |
|---|
| <p>* Step 2: Message SetDisplayMessageResponse - status <i>Accepted</i></p> <p>* Step 4: Message SetDisplayMessageResponse - status <i>Accepted</i></p> <p>* Step 6: Message: GetDisplayMessagesResponse - status <i>Accepted</i></p> <p>* Step 7: Message: NotifyDisplayMessagesRequest - requestId <Generated RequestId> - transactionId <Generated transactionId> - priority <i>InFront</i> - message.content <Configured message></p> <p>* Step 10: Message: GetDisplayMessagesResponse - status <i>Accepted</i></p> <p>* Step 11: Message: NotifyDisplayMessagesRequest - requestId <Generated RequestId> - transactionId <Generated transactionId> - priority <i>InFront</i> - message.content <Configured message with a " 2" extended to it.></p> <p>* Step 18: Message: GetDisplayMessagesResponse - status <i>Unknown</i></p> <p>* Step 20: Message: GetDisplayMessagesResponse - status <i>Unknown</i></p> |
| Post scenario validations: - N/a |

TC_O_32_CS: Set Display Message - Specific transaction - Second Alwaysfront priority

| | |
|--------------------------|--|
| Test case name | Set Display Message - Specific transaction - Second Alwaysfront priority |
| Test case Id | TC_O_32_CS |
| Use case Id(s) | O02 |
| Requirement(s) | O02_FR_18 |
| System under test | Charging Station |
| Description | This test case describes how a CSO can set a message to be displayed on a Charging Station for a specific transaction. Depending on the given parameters the message shall be displayed a certain way on the Charging Station. |
| Purpose | To verify if the Charging Station is able to display multiple additional messages with a "AlwaysFront" priority for a specific transaction according to the DisplayMessage mechanism as described in the OCPP specification. |
| Prerequisite(s) | N/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: State is <i>EnergyTransferStarted</i> |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a SetDisplayMessageResponse | 1. The Test System sends a SetDisplayMessageRequest with message.id <Generated displayMessageId> message.transactionId <Received transactionId> AND message.priority AlwaysFront AND message.display <Configured display component variable> (Omitted when left empty) |
| <u>Note(s):</u> - Display message <Generated displayMessageId> is shown | |
| 4. The Charging Station responds with a SetDisplayMessageResponse | 3. The Test System sends a SetDisplayMessageRequest with message.id <Configured displayMessage2Id> message.transactionId <Received transactionId> AND message.priority AlwaysFront AND message.display <Configured display component variable> (Omitted when left empty) |
| <u>Note(s):</u> - Display message <Generated displayMessage1Id> is not displayed anymore - Display message <Generated displayMessage2Id> is shown | |
| 6. The Charging Station responds with a GetDisplayMessagesResponse | 5. The Test System sends a GetDisplayMessagesRequest with id <Generated displayMessageId> |
| 8. The Charging Station responds with a GetDisplayMessagesResponse | 7. The Test System sends a GetDisplayMessagesRequest with id <Generated displayMessageId2> requestId <Generated requestId> |
| 9. The Charging Station sends a NotifyDisplayMessagesRequest | 10. The Test System responds with a NotifyDisplayMessagesResponse . |
| 11. Execute Reusable State <i>StopAuthorized</i> | |
| 12. Execute Reusable State <i>EVConnectedPostSession</i> | |

| | |
|---|---|
| Main (Test scenario) | |
| 13. Execute Reusable State <i>EVDisconnected</i> | |
| 14. Execute Reusable State <i>ParkingBayUnoccupied</i> | |
| Note(s): - Display message <Generated displayMessage2Id> is not displayed anymore | |
| 16. The Charging Station responds with a GetDisplayMessagesResponse | 15. The Test System sends a GetDisplayMessagesRequest with id <Generated displayMessageId2> |

| |
|--|
| Tool validations |
| * Step 2: Message SetDisplayMessageResponse - status <i>Accepted</i> * Step 4: Message SetDisplayMessageResponse - status <i>Accepted</i> * Step 6: Message GetDisplayMessagesResponse - status <i>Unknown</i> * Step 8: Message GetDisplayMessagesResponse - status <i>Accepted</i> * Step 9: Message: NotifyDisplayMessagesRequest - requestId <Generated RequestId> - transactionId <Generated transactionId> - priority <i>AlwaysFront</i> - message.content <Configured message with a "2" extended to it.> * Step 16: Message: GetDisplayMessagesResponse - status <i>Unknown</i> |
| Post scenario validations: - N/a |

TC_O_33_CS: Get a Specific Display Message - No DisplayMessages configured

| | |
|-------------------|--|
| Test case name | Get a Specific Display Message - No DisplayMessages configured |
| Test case Id | TC_O_33_CS |
| Use case Id(s) | 004 |
| Requirement(s) | 004_FR_07 |
| System under test | Charging Station |
| Description | This test case describes how a CSO can request specific installed DisplayMessages configured via OCPP in a Charging Station. The Charging Station can remove messages when they are out-dated, or transactions have ended. It can be very useful for a CSO to be able to view to current list of messages, so the CSO knows which messages are (still) configured. |
| Purpose | To verify if the Charging Station is able to respond correctly when a specific id message is requested by the CSMS but no messages are configured according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | N/a |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State:

N/a

Main (Test scenario)

| Charging Station | CSMS |
|---|---|
| 2. The Charging Station responds with a GetDisplayMessagesResponse | 1. The Test System sends a GetDisplayMessagesRequest with id <Generated displayMessageId> |

Tool validations

* Step 2:

Message **GetDisplayMessagesResponse**- **status** *Unknown*

Post scenario validations:

- N/a

TC_O_34_CS: Get a Specific Display Message - Known Id, but not matching State

| | |
|-------------------|--|
| Test case name | Get a Specific Display Message - Known Id, but not matching State |
| Test case Id | TC_O_34_CS |
| Use case Id(s) | O04 |
| Requirement(s) | O04_FR_02 |
| System under test | Charging Station |
| Description | This test case describes how a CSO can request specific installed DisplayMessages configured via OCPP in a Charging Station. The Charging Station can remove messages when they are out-dated, or transactions have ended. It can be very useful for a CSO to be able to view to current list of messages, so the CSO knows which messages are (still) configured. |
| Purpose | To verify if the Charging Station is able to respond correctly when a specific id message is requested by the CSMS but the requested State is different according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | Configured display message state 1, must be different than display message state 2. |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: SetDisplayMessage with state <Configured Message State> |
| Reusable State: N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a GetDisplayMessagesResponse | 1. The Test System sends a GetDisplayMessagesRequest with id <Generated displayMessageId> AND state <Configured Message State 2> |

| |
|--|
| Tool validations |
| * Step 2: Message GetDisplayMessagesResponse - status <i>Unknown</i> |
| Post scenario validations: - N/a |

TC_O_35_CS: Get a Specific Display Message - Known Id, but not matching Priority

| | |
|-------------------|--|
| Test case name | Get a Specific Display Message - Known Id, but not matching Priority |
| Test case Id | TC_O_35_CS |
| Use case Id(s) | O04 |
| Requirement(s) | O04_FR_02 |
| System under test | Charging Station |
| Description | This test case describes how a CSO can request specific installed DisplayMessages configured via OCPP in a Charging Station. The Charging Station can remove messages when they are out-dated, or transactions have ended. It can be very useful for a CSO to be able to view to current list of messages, so the CSO knows which messages are (still) configured. |
| Purpose | To verify if the Charging Station is able to respond correctly when a specific id message is requested by the CSMS but the requested priority is different according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | Configured display message priority 1, must be different than display message priority 2. |

Before (Preparations)

Configuration State:

N/a

Memory State:

[SetDisplayMessage](#)

Reusable State:

N/a

Main (Test scenario)

| Charging Station | CSMS |
|---|--|
| 2. The Charging Station responds with a GetDisplayMessagesResponse | 1. The Test System sends a GetDisplayMessagesRequest with id <Generated displayMessageId> AND state <Configured Message Priority 2> |

Tool validations

* Step 2:

Message **GetDisplayMessagesResponse**- **status** *Unknown*

Post scenario validations:

- N/a

TC_O_36_CS: Set Display Message - State Charging

| | |
|-------------------|---|
| Test case name | Set Display Message - State Charging |
| Test case Id | TC_O_36_CS |
| Use case Id(s) | 001 |
| Requirement(s) | N/a |
| System under test | Charging Station |
| Description | This test case describes how the CSMS can be requested to sent an SetDisplayMessageRequest to the charging station. Depending on the given parameters the message shall be displayed a certain way and at a certain moment on the Charging Station. These messages are displayed additionally on a Charging Station and are not part of the firmware. |
| Purpose | To verify if the Charging Station is able to display specific messages while the chargingState is Charging according to the DisplayMessage mechanism as described in the OCPP specification. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a SetDisplayMessageResponse | 1. The Test System sends a SetDisplayMessageRequest with message.id <Generated displayMessageId> message.priority <Configured Priority> message.state Charging message.display <Configured display component variable> (Omitted when left empty) |
| Note(s): The display message should NOT be displayed. | |
| 3. Execute Reusable State <i>ParkingBayOccupied</i> | |
| 4. Execute Reusable State <i>Authorized</i> | |
| 5. Execute Reusable State <i>EVConnectedPreSession</i> | |
| 6. Execute Reusable State <i>EnergyTransferStarted</i> | |
| Note(s): The display message should be displayed. | |
| 7. Execute Reusable State <i>StopAuthorized</i> | |
| 8. Execute Reusable State <i>EVConnectedPostSession</i> | |
| 9. Execute Reusable State <i>EVDisconnected</i> | |
| 10. Execute Reusable State <i>ParkingBayUnoccupied</i> | |
| Note(s): The display message should NOT be displayed. | |
| 12. The Charging Station responds with a GetDisplayMessagesResponse | 11. The Test System sends a GetDisplayMessagesRequest with id <Generated displayMessageId> requestId <Generated requestId> |
| 13. The Charging Station sends a NotifyDisplayMessagesRequest | 14. The Test System responds with a NotifyDisplayMessagesResponse . |
| Note(s): If <i>tbc</i> is True at Step 13 then step 13 and 14 will be repeated | |

| Tool validations |
|--|
| <div>* Step 2: Message SetDisplayMessageResponse - status <i>Accepted</i></div> <div>* Step 12: Message GetDisplayMessagesResponse - status <i>Accepted</i></div> <div>* Step 13: Message NotifyDisplayMessagesRequest - requestId <i><Generated requestId></i> - state <i>Charging</i></div> |
| <div>Post scenario validations: - N/a</div> |

TC_O_37_CS: Set Display Message - State Idle

| | |
|-------------------|---|
| Test case name | Set Display Message - State Idle |
| Test case Id | TC_O_37_CS |
| Use case Id(s) | 001 |
| Requirement(s) | N/a |
| System under test | Charging Station |
| Description | This test case describes how the CSMS can be requested to sent an SetDisplayMessageRequest to the charging station. Depending on the given parameters the message shall be displayed a certain way and at a certain moment on the Charging Station. These messages are displayed additionally on a Charging Station and are not part of the firmware. |
| Purpose | To verify if the Charging Station is able to display specific messages while the chargingState is Idle according to the DisplayMessage mechanism as described in the OCPP specification. |
| Prerequisite(s) | N/a |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State:

N/a

Main (Test scenario)

| Charging Station | CSMS |
|--|--|
| 2. The Charging Station responds with a SetDisplayMessageResponse | 1. The Test System sends a SetDisplayMessageRequest with message.id <Generated displayMessageId> message.priority <Configured Priority> message.state Idle message.display <Configured display component variable> (Omitted when left empty) |
| <u>Note(s)</u> : The display message should be displayed. | |
| 3. Execute Reusable State <i>ParkingBayOccupied</i> | |
| 4. Execute Reusable State <i>Authorized</i> | |
| 5. Execute Reusable State <i>EVConnectedPreSession</i> | |
| 6. Execute Reusable State <i>EnergyTransferStarted</i> | |
| <u>Note(s)</u> : The display message should NOT be displayed. | |
| 7. Execute Reusable State <i>StopAuthorized</i> | |
| 8. Execute Reusable State <i>EVConnectedPostSession</i> | |
| 9. Execute Reusable State <i>EVDisconnected</i> | |
| 10. Execute Reusable State <i>ParkingBayUnoccupied</i> | |
| <u>Note(s)</u> : The display message should be displayed. | |
| 12. The Charging Station responds with a GetDisplayMessagesResponse | 11. The Test System sends a GetDisplayMessagesRequest with id <Generated displayMessageId> requestId <Generated requestId> |
| 13. The Charging Station sends a NotifyDisplayMessagesRequest | 14. The Test System responds with a NotifyDisplayMessagesResponse . |
| <u>Note(s)</u> : If tbc is True at Step 13 then step 13 and 14 will be repeated | |

| Tool validations |
|--|
| <div>* Step 2: Message SetDisplayMessageResponse - status <i>Accepted</i></div> <div>* Step 12: Message GetDisplayMessagesResponse - status <i>Accepted</i></div> <div>* Step 13: Message NotifyDisplayMessagesRequest - requestId <i><Generated requestId></i> - state <i>Idle</i></div> |
| <div>Post scenario validations: - N/a</div> |

TC_O_38_CS: Set Display Message - State Unavailable

| | |
|-------------------|---|
| Test case name | Set Display Message - State Unavailable |
| Test case Id | TC_O_38_CS |
| Use case Id(s) | 001 |
| Requirement(s) | N/a |
| System under test | Charging Station |
| Description | This test case describes how the CSMS can be requested to sent an SetDisplayMessageRequest to the charging station. Depending on the given parameters the message shall be displayed a certain way and at a certain moment on the Charging Station. These messages are displayed additionally on a Charging Station and are not part of the firmware. |
| Purpose | To verify if the Charging Station is able to display specific messages while the chargingState is Unavailable according to the DisplayMessage mechanism as described in the OCPP specification. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a SetDisplayMessageResponse | 1. The Test System sends a SetDisplayMessageRequest with message.id <Generated displayMessageId> message.priority <Configured Priority> message.state Unavailable message.display <Configured display component variable> (Omitted when left empty) |
| Note(s): The display message should NOT be displayed. | |
| 3. Execute Reusable State Unavailable | |
| Note(s): The display message should be displayed. | |
| 5. The Charging Station responds with a ChangeAvailabilityResponse | 4. The Test System sends a ChangeAvailabilityRequest with operationalStatus Operative |
| 6. The Charging Station notifies the CSMS about the current state of all its connectors (and optionally also the Charging Station itself and all EVSE). | 7. The Test System responds accordingly. |
| Note(s): The display message should NOT be displayed. | |
| 9. The Charging Station responds with a GetDisplayMessagesResponse | 8. The Test System sends a GetDisplayMessagesRequest with id <Generated displayMessageId> requestId <Generated requestId> |
| 10. The Charging Station sends a NotifyDisplayMessagesRequest | 11. The Test System responds with a NotifyDisplayMessagesResponse . |
| Note(s): If tbc is True at Step 10 then step 10 and 11 will be repeated | |

| Tool validations |
|---|
| <div>* Step 2: Message SetDisplayMessageResponse - status <i>Accepted</i></div> <div>* Step 5: Message ChangeAvailabilityResponse - status <i>Accepted</i></div> <div>* Step 6: Message: StatusNotificationRequest - connectorStatus <i>Available</i> Message: NotifyEventRequest - eventData[0].trigger <i>Delta</i> - eventData[0].actualValue <i>"Available"</i> - eventData[0].component.name <i>"ChargingStation" / EVSE / Connector</i> - eventData[0].variable.name <i>"AvailabilityState"</i></div> <div>* Step 9: Message GetDisplayMessagesResponse - status <i>Accepted</i></div> <div>* Step 10: Message NotifyDisplayMessagesRequest - requestId <i><Generated requestId></i> - state <i>Unavailable</i></div> |
| <div>Post scenario validations: - N/a</div> |

TC_O_39_CS: Set Display Message - State Faulted

| | |
|-------------------|---|
| Test case name | Set Display Message - State Faulted |
| Test case Id | TC_O_39_CS |
| Use case Id(s) | 001 |
| Requirement(s) | N/a |
| System under test | Charging Station |
| Description | This test case describes how the CSMS can be requested to sent an SetDisplayMessageRequest to the charging station. Depending on the given parameters the message shall be displayed a certain way and at a certain moment on the Charging Station. These messages are displayed additionally on a Charging Station and are not part of the firmware. |
| Purpose | To verify if the Charging Station is able to display specific messages while the chargingState is Faulted according to the DisplayMessage mechanism as described in the OCPP specification. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a SetDisplayMessageResponse | 1. The Test System sends a SetDisplayMessageRequest with message.id <Generated displayMessageId> message.priority <Configured Priority> message.state Faulted message.message.content <Message indicating the Charging Station is in a Faulted state> message.display <Configured display component variable> (Omitted when left empty) |
| Note(s): The display message should NOT be displayed. | |
| Manual Action: Set the Charging Station to state Faulted. | |
| 3. The Charging Station notifies the CSMS about the status change of the Charging Station. | 4. The Test System responds accordingly. |
| Note(s): - Step 3/4 are used to detect if the Charging Station status has changed. - The display message should be displayed now. | |
| Manual Action: Set the Charging Station back to state Available. | |
| 5. The Charging Station notifies the CSMS about the status change of the Charging Station. | 6. The Test System responds accordingly. |
| Note(s): - Step 5/6 are used to detect if the Charging Station status has changed. - The display message should NOT be displayed anymore. | |
| 8. The Charging Station responds with a GetDisplayMessagesResponse | 7. The Test System sends a GetDisplayMessagesRequest with id <Generated displayMessageId> requestId <Generated requestId> |
| 9. The Charging Station sends a NotifyDisplayMessagesRequest | 10. The Test System responds with a NotifyDisplayMessagesResponse . |
| Note(s): If tbc is True at Step 9 then step 9 and 10 will be repeated | |

Tool validations

* Step 2:

Message **SetDisplayMessageResponse**

- **status** *Accepted*

* Step 3:

At least one of the following messages must be sent:

Message: **StatusNotificationRequest**

- **connectorStatus** *Faulted* or *Unavailable*

Message: **NotifyEventRequest**

- **eventData[0].trigger** must be *Delta*

- **eventData[0].actualValue** must be *Faulted* or *Unavailable*

- **eventData[0].component.name** must be *Connector*

- **eventData[0].variable.name** must be *AvailabilityState*

- **evse.id** *<not omitted>*

- **connector.id** *<not omitted>*

Message: **NotifyEventRequest**

- **eventData[0].trigger** must be *Delta*

- **eventData[0].actualValue** must be *Faulted*

- **eventData[0].component.name** must be *ChargingStation*

- **eventData[0].variable.name** must be *AvailabilityState*

- **evse.id** *<omitted>*

- **connector.id** *<omitted>*

* Step 5:

At least one of the following messages must be sent:

Message: **StatusNotificationRequest**

- **connectorStatus** *Available*

Message: **NotifyEventRequest**

- **eventData[0].trigger** must be *Delta*

- **eventData[0].actualValue** must be *Available*

- **eventData[0].component.name** must be *Connector*

- **eventData[0].variable.name** must be *AvailabilityState*

- **evse.id** *<not omitted>*

- **connector.id** *<not omitted>*

Message: **NotifyEventRequest**

- **eventData[0].trigger** must be *Delta*

- **eventData[0].actualValue** must be *Available*

- **eventData[0].component.name** must be *ChargingStation*

- **eventData[0].variable.name** must be *AvailabilityState*

* Step 8:

Message **GetDisplayMessagesResponse**

- **status** *Accepted*

* Step 9:

Message **NotifyDisplayMessagesRequest**

- **requestId** *<Generated requestId>*

- **state** *Faulted*

Post scenario validations:

- N/a

P Data Transfer

TC_P_01_CS: Data Transfer to the Charging Station - Rejected / Unknown VendorId / Unknown MessageId

| | |
|-------------------|--|
| Test case name | Data Transfer to the Charging Station - Rejected / Unknown VendorId / Unknown MessageId |
| Test case Id | TC_P_01_CS |
| Use case Id(s) | P01 |
| Requirement(s) | P01.FR.05, P01.FR.06 |
| System under test | Charging Station |
| Description | The DataTransfer message to send information for functions that are not supported by OCPP. |
| Purpose | To verify whether the Charging Station is able to handle receiving a DataTransferRequest, even if it does not support any vendor-specific implementations. |
| Prerequisite(s) | The configured vendorId should not be implemented and the configured messageId should be unused. |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a DataTransferResponse | 1. The Test System sends a DataTransferRequest with vendorId <i>org.openchargealliance.Test System</i> messageId <i><Configured messageId></i> |

| |
|--|
| Tool validations |
| * Step 2: Message: DataTransferResponse - status must be <i>UnknownVendorId</i> OR <i>UnknownMessageId</i> OR <i>Rejected</i> (Rejected will also be allowed, because there are implementers that like to just reject the message when the Charging Station does not support any vendor-specific features. |
| Post scenario validations: N/a |

TC_P_03_CS: CustomData - Receive custom data

| | |
|-------------------|---|
| Test case name | CustomData - Receive custom data |
| Test case Id | TC_P_03_CS |
| Use case Id(s) | N/a |
| Requirement(s) | N/a |
| System under test | Charging Station |
| Description | Checks if the CS is able to receive custom data. |
| Purpose | To verify whether the CS is able to handle receiving custom data. |
| Prerequisite(s) | N/a |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with SetVariablesResponse | 1. Test System sends SetVariablesRequest with: - variable.name = "OfflineThreshold" - component.name = "OCPPCommCtrlr" - attributeValue = "200" - attributeType is Actual |
| 4. The Charging Station responds with GetVariablesResponse | 3. Test System sends GetVariablesRequest with: - variable.name = "OfflineThreshold" - component.name = "OCPPCommCtrlr" - attributeType is Actual |

| |
|---|
| Tool validations |
| * Step 2: Message: SetVariablesResponse - setVariableResult[0].attributeStatus <i>Accepted</i> * Step 4: Message: GetVariablesResponse - getVariableResult[0].attributeStatus <i>Accepted</i> - getVariableResult[0].attributeType <i>Actual</i> or omitted - getVariableResult[0].attributeValue <i>200</i> |
| Post scenario validations: - N/a |

TC_P_04_CS: Able to receive customData - ChargingProfile

| | |
|-------------------|--|
| Test case name | Able to receive customData - ChargingProfile |
| Test case Id | TC_P_04_CS |
| Use case Id(s) | N/a |
| Requirement(s) | N/a |
| System under test | Charging Station |
| Description | Checks if the CS is able to receive custom data. |
| Purpose | To verify whether the CS is able to handle receive custom data in smart charging profiles. |
| Prerequisite(s) | The Charging Station supports Smart Charging |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a SetChargingProfileResponse | <p>1. The Test System sends a SetChargingProfileRequest with</p> <p>evseld <Configured evseld></p> <p>chargingProfile.id <Configured chargingProfileId></p> <p>chargingProfile.chargingProfilePurpose TxDefaultProfile</p> <p>chargingProfile.customData <CustomData></p> <p>chargingProfile.chargingSchedule.duration <Configured duration></p> <p>chargingProfile.chargingSchedule.chargingRateUnit <Configured chargingRateUnit></p> <p>chargingProfile.chargingSchedule.chargingSchedulePeriod.startPeriod 0</p> <p>chargingProfile.chargingSchedule.chargingSchedulePeriod.limit if unit is A then 6(A) else 6000(W)</p> <p>chargingProfile.chargingSchedule.chargingSchedulePeriod.numberPhases <Configured numberPhases></p> <p>chargingProfile.chargingSchedule.chargingSchedulePeriod.customData <CustomData></p> |

| |
|---|
| Tool validations |
| <p>* Step 2:</p> <p>Message SetChargingProfileResponse</p> <p>- status Accepted</p> |
| <p>Post scenario validations:</p> <p>- N/a</p> |

Memory states

TransactionEventsInQueueEnded

| | |
|--------------------------|---|
| State | TransactionEventsInQueueEnded |
| System under test | Charging Station |
| Description | This state will prepare the Charging Station, so that there will be TransactionEventRequests stored in its queue from an ended Transaction. |

| |
|---|
| Before (Preparations) |
| Configuration State: OfflineTxForUnknownIdEnabled is <i>true</i> (If implemented) |
| Memory State: <i>IdTokenCached</i> for <Configured valid IdToken fields> (If implemented) <i>IdTokenLocalAuthList</i> for <Configured valid IdToken fields> (If implemented) |
| Reusable State(s): N/a |

| | |
|---|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Test System closes the WebSocket connection AND does not accept a reconnect. | |
| <u>Manual Action</u> : Drive EV into parking bay. | |
| <u>Manual Action</u> : Connect the EV and EVSE. | |
| <u>Manual Action</u> : Present idToken. | |
| <u>Manual Action</u> : Present the same idToken as used to start the transaction. | |
| <u>Manual Action</u> : Disconnect the EV and EVSE. | |
| <u>Manual Action</u> : Drive EV out of parking bay. | |
| 2. The Test System accepts reconnection attempt from the Charging Station. | |

| |
|--|
| Tool validations |
| N/a |
| Post scenario validations: TransactionEventRequest messages are stored in the queue of the Charging Station. |

CertificateInstalled

| | |
|--------------------------|--|
| State | CertificateInstalled |
| System under test | Charging Station |
| Description | A pre configured certificate of the specified certificateType will be installed. |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a InstallCertificateResponse | 1. The Test System sends a InstallCertificateRequest with certificateType is <Specified certificateType> certificate is <Corresponding certificate> |

| |
|--|
| Tool validations |
| * Step 2: Message: InstallCertificateResponse - status must be <i>Accepted</i> |
| Post scenario validations: Certificate of the specified certificateType is stored at the Charging Station. |

IdTokenCached

| | |
|--------------------------|--|
| State | IdTokenCached |
| System under test | Charging Station |
| Description | An idToken is stored in the Authorization Cache of the Charging Station. |

Before (Preparations)

| |
|--|
| Configuration State: - AuthCacheCtrlr.Enabled is <i>true</i> (If implemented) |
| Memory State: N/a |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|--|-------------|
| Charging Station | CSMS |
| 1. Execute Reusable State <i>ParkingBayoccupied</i> | |
| 2. Execute Reusable State <i>Authorized</i> | |
| <u>Note(s)</u> : Step 3 and onwards are executed in case the idToken at step 2 was Accepted. | |
| 3. Execute Reusable State <i>EVConnectedPreSession</i> | |
| 4. Execute Reusable State <i>EnergyTransferStarted</i> | |
| 5. Execute Reusable State <i>StopAuthorized</i> | |
| 6. Execute Reusable State <i>EVDisconnected</i> | |
| 7. Execute Reusable State <i>ParkingBayUnoccupied</i> | |

| Tool validations |
|------------------|
| N/a |

IdTokenCached15118

| | |
|-------------------|---|
| State | IdTokenCached15118 |
| System under test | Charging Station |
| Description | A 15118-idToken is stored in the Authorization Cache of the Charging Station. |

| |
|---|
| Before (Preparations) |
| Configuration State: - AuthCacheCtrlr.Enabled is true (If implemented) |
| Memory State: N/a |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|--|------|
| Charging Station | CSMS |
| 1. Execute Reusable State <i>ParkingBayoccupied</i> | |
| 2. Execute Reusable State <i>EVConnectedPreSession</i> | |
| 3. Execute Reusable State <i>Authorized15118</i> | |
| 4. Execute Reusable State <i>EnergyTransferStarted</i> | |
| 5. Execute Reusable State <i>StopAuthorized</i> (Remote) | |
| 6. Execute Reusable State <i>EVDisconnected</i> | |
| 7. Execute Reusable State <i>ParkingBayUnoccupied</i> | |

| |
|------------------|
| Tool validations |
| N/a |

IdTokenLocalAuthList

| | |
|-------------------|---|
| State | IdTokenLocalAuthList |
| System under test | Charging Station |
| Description | An valid idToken is stored in the Local Authorization List of the Charging Station. |

| |
|---|
| Before (Preparations) |
| Configuration State: LocalAuthListCtrlr.Enabled is true (If implemented) |
| Memory State: N/a |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|--|--|
| Charging Station | CSMS |
| 2. The Charging Station responds with a SendLocalListResponse | 1. The Test System sends a SendLocalListRequest with updateType Full localAuthorizationList[0].idToken.idToken <Configured <i>valid_idtoken_idtoken</i> > localAuthorizationList[0].idToken.type <Configured <i>valid_idtoken_type</i> > |

| |
|---|
| Tool validations |
| * Step 2: (Message: SendLocalListResponse) status is Accepted |
| Post scenario validations: N/a |

SetChargingProfile

| | |
|--------------------------|---|
| State | SetChargingProfile |
| System under test | Charging Station |
| Description | This will store a Charging Profile at the Charging Station. |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State(s):

N/a

Main (Test scenario)

| | |
|---|--|
| Charging Station | CSMS |
| 2. The Charging Station responds with a SetChargingProfileResponse | 1. The Test System sends a SetChargingProfileRequest with chargingProfile <Provided chargingProfile> |

Tool validations

* Step 2:

(Message: **SetChargingProfileResponse**)**status** is *Accepted*

Post scenario validations:

N/a

SetDisplayMessage

| | |
|-------------------|--|
| State | SetDisplayMessage |
| System under test | Charging Station |
| Description | This will set a display message at the Charging Station. |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a SetDisplayMessageResponse | 1. The Test System sends a SetDisplayMessageRequest with message.id <Generated displayMessageId> message.priority <Configured priority> message.state <Omitted, unless specifically described at the testcase> message.display <Configured display component variable> (Omitted when left empty) message.message.format <Configured Message Format> message.message.content <Configured Message> |

| |
|---|
| Tool validations |
| * Step 2: (Message: SetDisplayMessageResponse) status is Accepted |
| Post scenario validations: N/a |

RenewChargingStationCertificate

| | |
|--------------------------|---|
| State | RenewChargingStationCertificate |
| System under test | Charging Station |
| Description | The ChargingStationCertificate is renewed using A02/A03 |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State(s):

N/a

Main (Test scenario)

| Charging Station | CSMS |
|--|---|
| 2. The Charging Station responds with a TriggerMessageResponse | 1. The Test System sends a TriggerMessageRequest With requestedMessage <i>SignChargingStationCertificate</i> |
| 3 The Charging Station sends a SignCertificateRequest | 4. The Test System responds with a SignCertificateResponse With status <i>Accepted</i> |
| 6. The Charging Station responds with a CertificateSignedResponse | 5. The Test System sends a CertificateSignedRequest With certificateChain <i><Certificate generated from the received CSR from step 3 and signed by the provided CSMS Root certificate></i> certificateType <i>ChargingStationCertificate</i> |
| <i>If the certificate is valid, then Charging Station should reconnect with the new certificate. Test System waits some time for a reconnection, and if that does not occur, will drop the connection to force a reconnection.</i> | |
| 7. The Charging Station reconnects. | |
| 8. If Charging Station rebooted: The Charging Station sends a BootNotificationRequest | 9. Test System responds with a BootNotificationResponse . |

Tool validations

* Step 2:

Message: **TriggerMessageResponse**- **status** must be *Accepted*

* Step 3:

Message: **SignCertificateRequest**- **csr** must contain *<An CSR that meets the following requirements:**When using RSA or DSA the key must be at least 2048 bits long.**and when using elliptic curve cryptography the key must be at least 224 bits long.**The received CSR must be transmitted as described in RFC 2986 and then encoded in Privacy-Enhanced Mail (PEM) format.>*

* Step 6:

Message: **CertificateSignedResponse**- **status** must be *Accepted*

* Step 7:

Charging Station must reconnect with new certificate.

Post scenario validations:

N/a

RenewV2GChargingStationCertificate

| | |
|--------------------------|---|
| State | RenewV2GChargingStationCertificate |
| System under test | Charging Station |
| Description | The V2G ChargingStationCertificate is renewed using A02/A03 |

Before (Preparations)

Configuration State:

ISO15118Ctrlr.V2GCertificateInstallationEnabled is *true* if implemented

ISO15118Ctrlr.CountryName is *NL* if implemented

ISO15118Ctrlr.OrganizationName is configured vendorId if implemented

Test System will check all configured **ISO15118Ctrlr.SecclId**'s using a **GetBaseReportRequest**

Memory State:

N/a

Reusable State(s):

N/a

Main (Test scenario)

| Charging Station | CSMS |
|--|---|
| 2. The Charging Station responds with a TriggerMessageResponse | 1. The Test System sends a TriggerMessageRequest With requestedMessage <i>SignV2GCertificate</i> EVSE EVSE (having an <i>secclId</i>) returned in the <i>GetReportResponse</i> or omitted in case none is available |
| 3 The Charging Station sends a SignCertificateRequest | 4. The Test System responds with a SignCertificateResponse With status <i>Accepted</i> |
| 6. The Charging Station responds with a CertificateSignedResponse | 5. The Test System sends a CertificateSignedRequest With certificateChain <Certificate generated from the received CSR from step 3 and signed by SubCA2 or SubCA (if SubCA2 does not exist) certificate from the provided V2G certificate chain> certificateType <i>V2GCertificate</i> |

Note(s): Steps 1, 2, 3, 4, 5, and 6 are repeated for all returned *secclId*s

Tool validations

* Step 2:

Message: **TriggerMessageResponse**

- **status** must be *Accepted*

* Step 3:

Message: **SignCertificateRequest**

- **csr** must contain <An CSR that meets the following requirements:

The key must be at least 256 bits long.

The received CSR must be transmitted as described in RFC 2986 and then encoded in Privacy-Enhanced Mail (PEM) format.>

The certificate can only be an ECDSA certificate (ISO15118 cannot be used with RSA).

If an *secclId* is found the *csr* should contain the *secclId* in the CN.

* Step 6:

Message: **CertificateSignedResponse**

- **status** must be *Accepted*

Post scenario validations:

N/a

Reusable states

Testcases can refer to a reusable state at the before or main stage. The steps described at the reusable state will be executed and then it will return to the testcase that called the reusable state.

Booting

| | |
|--------------------------|---|
| State | Booting |
| System under test | Charging Station |
| Description | This state will prepare the Charging Station, so that it is still booting. The connection has not been setup yet. |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a ResetResponse | 1. The Test System sends a ResetRequest with type Immediate |

| |
|---|
| Tool validations |
| * Step 2: Message: ResetResponse - status must be <i>Accepted</i> |
| Post scenario validations: State is <i>Booting</i> |

Booted

| | |
|--------------------------|--|
| State | Booted |
| System under test | Charging Station |
| Description | This state will reset or power cycle the Charging Station, depending on the testcase. The charging station ends in a state where it is booted back up and is in idle mode. |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State(s):

N/a

Main (Test scenario)

| | |
|---|--|
| Charging Station | CSMS |
| Manual Action: <i>Power cycle the Charging Station.</i> OR execute step 1 and 2, depending on the testcase. | |
| 2. The Charging Station responds with a ResetResponse with status Accepted | 1. The Test System sends a ResetRequest |
| 3. The Charging Station sends a BootNotificationRequest | 4. The Test System responds with a BootNotificationResponse with status Accepted |
| 5. The Charging Station notifies the CSMS about the current state of all connectors. | 6. The Test System responds accordingly. |
| 7 The Charging Station sends a SecurityEventNotificationRequest | 8 The Test System responds with a SecurityEventNotificationResponse |

Tool validations

* Step 2:

Message: **ResetResponse**- **status Accepted**

* Step 5:

Message: **StatusNotificationRequest**- **connectorStatus Available**- **evseld** not 0- **connectorId** not 0Message: **NotifyEventRequest**- **eventData[0].trigger Delta**- **eventData[0].actualValue "Available"**- **eventData[0].component.name "Connector"**- **eventData[0].variable.name "AvailabilityState"**

* Step 7:

Message: **SecurityEventNotificationRequest**- **type** must be *StartupOfTheDevice* OR *ResetOrReboot*

Post scenario validations:

State is *Booted*

Reserved

| | |
|--------------------------|---|
| State | Reserved |
| System under test | Charging Station |
| Description | This state will prepare the Charging Station, so that one of its EVSE becomes reserved. |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State(s):

N/a

Main (Test scenario)

| Charging Station | CSMS |
|---|---|
| 2. The Charging Station responds with a ReserveNowResponse | 1. The Test System sends a ReserveNowRequest with evseld is <Specified evseld (Configured evseld as a default)> idToken.idToken <Specified valid_idtoken_idtoken (Configured idToken as a default)> idToken.type <Specified valid_idtoken_type> |
| 3. The Charging Station notifies the CSMS about the status change of the connector. <u>Note(s):</u> - The Test System expects that the Charging Station sets the <i>availabilityState</i> of the EVSE and corresponding connectors to <i>Reserved</i> . - Reporting the <i>AvailabilityState</i> of the EVSE component itself is optional. | 4. The Test System responds accordingly. |

Tool validations

* Step 2:

Message: **ReserveNowResponse**- **status** must be *Accepted*

* Step 3:

Message: **StatusNotificationRequest**- **evseld** not 0- **connectorId** not 0- **connectorStatus** must be *Reserved*Message: **NotifyEventRequest**- **eventData[0].trigger** must be *Delta*- **eventData[0].actualValue** must be *Reserved*- **eventData[0].component.name** must be *Connector*- **eventData[0].evse.id** not 0- **eventData[0].evse.connectorId** not 0- **eventData[0].variable.name** must be *AvailabilityState*

(Optional)

Message: **NotifyEventRequest**- **eventData[0].trigger** must be *Delta*- **eventData[0].actualValue** must be *Reserved*- **eventData[0].component.name** must be *EVSE*- **eventData[0].variable.name** must be *AvailabilityState*

Post scenario validations:

State is *Reserved*

Unavailable

| | |
|--------------------------|---|
| State | Unavailable |
| System under test | Charging Station |
| Description | This state will prepare the Charging Station, so that the Charging Station / EVSEs / connectors are set to AvailabilityState Unavailable. |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Charging Station responds with a ChangeAvailabilityResponse | 1. The Test System sends a ChangeAvailabilityRequest with operationalStatus <i>Inoperative</i> evse.id <Specified evseld> evse.connectorId <Specified connectorId> |
| 3. The Charging Station notifies the CSMS about the current state of all connectors belonging to the specified component(s). | 4. The Test System responds accordingly. |

| |
|---|
| Tool validations |
| <p>* Step 2: Message ChangeAvailabilityResponse - status <i>Accepted</i></p> <p>* Step 3: Message: StatusNotificationRequest - connectorStatus <i>Unavailable</i> - evseld <Specified evseld> - connectorId <Specified connectorId></p> <p>Message: NotifyEventRequest - eventData[0].trigger <i>Delta</i> - eventData[0].actualValue <i>"Unavailable"</i> - eventData[0].component.name <i>"ChargingStation" / EVSE / Connector</i> - eventData[0].variable.name <i>"AvailabilityState"</i></p> |
| <p>Post scenario validations: State is <i>Reserved</i></p> |

ParkingBayOccupied

| | |
|--------------------------|---|
| State | ParkingBayOccupied |
| System under test | Charging Station |
| Description | This state will prepare the Charging Station, so that the EV entered the parking bay. The execution of this State is optional. Because there may not be a parking bay occupancy sensor OR the Charging Station is being tested with a test plug or EV Simulator. |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action:</u> Drive EV into parking bay. | |
| <u>Note(s):</u> - This State is optional (Even when TxStartPoint contains ParkingBayOccupancy). | |
| 1. The Charging Station sends a TransactionEventRequest <u>Note(s):</u> - This step needs to be executed when TxStartPoint contains ParkingBayOccupancy AND the EV entered the parking bay. | 2. The Test System responds with a TransactionEventResponse |

| |
|---|
| Tool validations |
| * Step 1: Message: TransactionEventRequest - triggerReason must be EVDetected |
| Post scenario validations: State is ParkingBayOccupied |

EVConnectedPreSession

| | |
|--------------------------|--|
| State | EVConnectedPreSession |
| System under test | Charging Station |
| Description | This state will prepare the Charging Station, so that the EV and EVSE are connected. |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State(s):

If **State** is NOT [ParkingBayOccupied](#) then execute **Reusable State** [ParkingBayOccupied](#)

Main (Test scenario)

| Charging Station | CSMS |
|--|--|
| <u>Manual Action:</u> <i>Connect the EV and EVSE.</i> | |
| 1. The Charging Station notifies the CSMS about the status change of the connector. | 2. The Test System responds accordingly. |
| 3. The Charging Station sends a TransactionEventRequest | 4. The Test System responds with a TransactionEventResponse |
| <u>Note(s):</u> - This step is executed with eventType Started if: TxStartPoint contains EVConnected OR TxStartPoint contains PowerPathClosed and the test case was initiated from the state Authorized OR - This step is executed with eventType Updated if: TxStartPoint contains ParkingBayOccupancy OR TxStartPoint contains Authorized and the test case was initiated from the state Authorized . | |

Tool validations

* Step 1:

Message: **StatusNotificationRequest**

- **evseld** <configured evseld>
- for the connector involved in the transaction: **connectorStatus** *Occupied*
- optionally for other connectors of the same EVSE: **connectorStatus** *Available* or *Unavailable*

Message: **NotifyEventRequest**

- **eventData[0].trigger** must be *Delta*
- **eventData[0].component.name** must be *Connector*
- **eventData[0].variable.name** must be *AvailabilityState*
- **evse.id** <configured evseld>
- for the connector involved in the transaction: **eventData[0].actualValue** *Occupied*
- optionally for other connectors of the same EVSE: **eventData[0].actualValue** *Available* or *Unavailable*

* Step 3:

Message: **TransactionEventRequest**

- **eventType** started if **TxStartPoint** is EVConnected or PowerPathClosed and **State** is [Authorized](#), else updated
- **triggerReason** must be *CablePluggedIn* or *ChargingStateChanged* or *RemoteStart*
- **transactionInfo.chargingState** must be EVConnected or SuspendedEVSE or Charging if **State** is [Authorized](#)
- **evse.id** <configured evseld>
- **connector.id** <configured connectorId>

Post scenario validations:

State is EVConnectedPreSession

Authorized

| State | Authorized |
|-------------------|---|
| System under test | Charging Station |
| Description | <p>This state will prepare the Charging Station, so that the transaction is authorized. This can be done in two ways (The default way is configurable at Test System. This will be used when the calling testcase does not define which one to use.):</p> <p>A. Using local authorization</p> <p>B. Using a RequestStartTransactionRequest</p> |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): If State is NOT ParkingBayOccupied OR EVConnectedPreSession , then execute Reusable State ParkingBayOccupied |

| | |
|--|---|
| Main A (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action:</u> Present <i>idToken</i> . | |
| <p>1. The Charging Station sends an AuthorizeRequest</p> <p><u>Note(s):</u> - This step needs to be executed, unless (AuthEnabled is implemented with mutability <i>ReadOnly</i> AND the value is set to <i>false</i>) OR a start button as described at Use case C02 is used (This must be configured at the Test System) OR the idToken is cached. In case the idToken is used for a reservation, sending the AuthorizeRequest message is optional.</p> | <p>2. The Test System responds with an AuthorizeResponse with idTokenInfo.status <i>Accepted</i></p> |
| <p>3. The Charging Station sends a TransactionEventRequest</p> <p><u>Note(s):</u> - This step is executed with eventType <i>Started</i> if: TxStartPoint contains <i>Authorized</i> OR TxStartPoint contains <i>PowerPathClosed</i> and the test case was initiated from the state EVConnectedPreSession OR - This step is executed with eventType <i>Updated</i> if: TxStartPoint contains <i>ParkingBayOccupancy</i> OR TxStartPoint contains <i>EVConnected</i> and the test case was initiated from the state EVConnectedPreSession.</p> | <p>4. The Test System responds with a TransactionEventResponse</p> <p><u>Note(s):</u> - The first TransactionEventRequest sent after authorization contains the idToken field, unless a Start button was used to start the transaction. In case there is an idToken used, the TransactionEventResponse of this request message contains idTokenInfo with status <i>Accepted</i></p> |

| |
|--|
| Tool validations |
| <p>* Step 1: Message: AuthorizeRequest - idToken.idToken <Configured <i>valid_idtoken_idtoken</i>> - idToken.type <Configured <i>valid_idtoken_type</i>></p> <p>* Step 3: Message: TransactionEventRequest - triggerReason must be <i>Authorized</i> - idToken.idToken <Configured <i>valid_idtoken_idtoken</i>> - idToken.type <Configured <i>valid_idtoken_type</i>></p> |

| Main B (Test scenario) | |
|---|--|
| Charging Station | CSMS |
| 2. The Charging Station responds with a RequestStartTransactionResponse | 1. The Test System sends a RequestStartTransactionRequest with idToken.idToken <Configured valid_idtoken_idtoken> idToken.type <Configured valid_idtoken_type> evseld <Configured evseld> |
| 3. The Charging Station sends an AuthorizeRequest <u>Note(s):</u> - This step needs to be executed when AuthCtrlr.AuthorizeRemoteStart is true, unless (AuthEnabled is implemented with mutability ReadOnly AND the value is set to false) OR the idToken is cached. In case the idToken is used for a reservation, sending the AuthorizeRequest message is optional. | 4. The Test System responds with an AuthorizeResponse with idTokenInfo.status Accepted |
| 5. The Charging Station sends a TransactionEventRequest <u>Note(s):</u> - This step is executed with eventType Started if: TxStartPoint contains Authorized OR TxStartPoint contains PowerPathClosed and the test case was initiated from the state EVConnectedPreSession OR - This step is executed with eventType Updated if: TxStartPoint contains ParkingBayOccupancy OR TxStartPoint contains EVConnected and the test case was initiated from the state EVConnectedPreSession . | 6. The Test System responds with a TransactionEventResponse <u>Note(s):</u> - The first TransactionEventRequest sent after authorization contains the idToken field. The TransactionEventResponse of this request message contains idTokenInfo with status Accepted |

| Tool validations |
|--|
| <p>* Step 2: Message: RequestStartTransactionResponse - status must be Accepted If the transaction has already been started, so if TxStartPoint contains <i>ParkingBayOccupancy</i> OR (<Configured TxStartPoint> contains <i>EVConnected</i> AND State pre reusable state execution was <i>EVConnectedPreSession</i>) then - transactionId must be <Provided transactionId in first TransactionEventRequest></p> <p>* Step 3: Message: AuthorizeRequest - idToken.idToken <Configured valid_idtoken_idtoken> - idToken.type <Configured valid_idtoken_type></p> <p>* Step 5: Message: TransactionEventRequest - eventType Started if TxStartPoint is Authorized or PowerPathClosed and State is <i>EVConnectedPreSession</i>, else updated - triggerReason must be RemoteStart - transactionInfo.remoteStartId must be present. - idToken.idToken <Configured valid_idtoken_idtoken> - idToken.type <Configured valid_idtoken_type></p> <p>Post scenario validations: State is Authorized</p> |

Authorized15118

| | |
|--------------------------|---|
| State | Authorized15118 |
| System under test | Charging Station |
| Description | This state will prepare the Charging Station, so that the transaction is authorized, using plug and charge. |
| Prerequisite | State is EVConnectedPreSession |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Charging Station sends an <code>AuthorizeRequest</code> <u>Note(s):</u> <i>-The test case should be robust enough to also handle a <code>GetCertificateStatusRequest</code> or <code>Get15118EVCertificateRequest</code> and then expect the <code>AuthorizeRequest</code>.</i> | 2. The Test System responds with an <code>AuthorizeResponse</code> with <code>idTokenInfo.status</code> <i>Accepted</i> and <code>certificateStatus</code> <i>Accepted</i> |
| 3. The Charging Station sends a <code>TransactionEventRequest</code> <u>Note(s):</u> <i>- This step is executed with <code>eventType</code> Started if: <code>TxStartPoint</code> contains <i>Authorized</i> OR <code>TxStartPoint</code> contains <i>PowerPathClosed</i> (prerequisite state EVConnectedPreSession)</i> OR <i>- This step is executed with <code>eventType</code> Updated if: <code>TxStartPoint</code> contains <i>ParkingBayOccupancy</i> OR <code>TxStartPoint</code> contains <i>EVConnected</i> (prerequisite state EVConnectedPreSession)</i> | 4. The Test System responds with a <code>TransactionEventResponse</code> <u>Note(s):</u> <i>- The first <code>TransactionEventRequest</code> sent after authorization contains the <code>idToken</code> field.</i> |
| 5. The Charging Station sends a <code>NotifyEVChargingNeedsRequest</code> | 6. Test System responds with a <code>NotifyEVChargingNeedsResponse</code> with <code>status</code> <i>Accepted</i> |

| |
|--|
| Tool validations |
| <p>* Step 1: Message: <code>AuthorizeRequest</code> - <code>idToken.type</code> <i>eMAID</i> - <code>iso15118CertificateHashData</code> <i><Not omitted></i> If <code>ISO15118Ctrlr.CentralContractValidationAllowed</code> is <i>true</i>: - <code>certificate</code> <i><Not omitted></i> If <code>ISO15118Ctrlr.CentralContractValidationAllowed</code> is <i>false</i>: - <code>certificate</code> <i><Omitted></i></p> <p>* Step 3: Message: <code>TransactionEventRequest</code> - <code>triggerReason</code> must be <i>Authorized</i> - <code>idToken</code> <i><Not omitted></i> - <code>transactionInfo.transactionId</code> must be <i><transactionId></i></p> |

EnergyTransferStarted

| | |
|--------------------------|--|
| State | EnergyTransferStarted |
| System under test | Charging Station |
| Description | This state will prepare the Charging Station, so that the Charging Station is transferring energy between the EV and EVSE. |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State(s):

If **State** is NOT [Authorized](#) then execute **Reusable State** [Authorized](#)

If connector <configured connectorId> state is not occupied, then execute **Reusable State** [EVConnectedPreSession](#)

Main (Test scenario)

| Charging Station | CSMS |
|--|--|
| 1. The Charging Station sends a TransactionEventRequest <u>Note(s):</u> - This step is only executed if TxStartPoint is DataSigned, in which case the eventType will be Started. | 2. The Test System responds with a TransactionEventResponse |
| 3. The Charging Station sends a TransactionEventRequest <u>Note(s):</u> - Step 3 and 4 are optional. | 4. The Test System responds with a TransactionEventResponse |
| 5. The Charging Station sends a TransactionEventRequest <u>Note(s):</u> - This step is executed with eventType Started if: TxStartPoint contains EnergyTransfer OR - This step is executed with eventType Updated if: TxStartPoint contains ParkingBayOccupancy OR TxStartPoint contains Authorized OR TxStartPoint contains EVConnected OR TxStartPoint contains PowerPathClosed OR TxStartPoint contains DataSigned | 6. The Test System responds with a TransactionEventResponse |

Tool validations

* Step 1:

Message: **TransactionEventRequest**

- **triggerReason** must be SignedDataReceived

* Step 3:

Message: **TransactionEventRequest**

- **triggerReason** must be ChargingStateChanged

- **transactionInfo.chargingState** must be SuspendedEVSE

* Step 5:

Message: **TransactionEventRequest**

- **triggerReason** must be ChargingStateChanged

- **transactionInfo.chargingState** must be Charging

| |
|---|
| Tool validations |
| Post scenario validations: State is <i>EnergyTransferStarted</i> |

EnergyTransferSuspended

| | |
|--------------------------|--|
| State | EnergyTransferSuspended |
| System under test | Charging Station |
| Description | This state will prepare the Charging Station, so that it is in a state where the energy transfer is suspended by the EV. |
| Prerequisite | N/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): If State is NOT EnergyTransferStarted then execute Reusable State EnergyTransferStarted |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Notes(s)</u> : The tool will wait for <Configured Transaction Duration> seconds | |
| <u>Manual Action</u> : The EV suspends the energy transfer. | |
| 1. The Charging Station sends a TransactionEventRequest | 2. The Test System responds with a TransactionEventResponse |

| |
|---|
| Tool validations |
| <p>* Step 1:</p> <p>Message: TransactionEventRequest</p> <p>IF TxStopPoint contains <i>EnergyTransfer</i> THEN:</p> <ul style="list-style-type: none"> - triggerReason must be <i>ChargingStateChanged</i> (If chargingState = <i>SuspendedEV</i>) - transactionInfo.chargingState must be <i>EVConnected</i> OR <i>SuspendedEV</i> - transactionInfo.stoppedReason must be <i>StoppedByEV</i> - eventType must be <i>Ended</i> <p>ELSE:</p> <ul style="list-style-type: none"> - triggerReason must be <i>ChargingStateChanged</i> - transactionInfo.chargingState must be <i>SuspendedEV</i> - eventType must be <i>Updated</i> <p>END</p> |
| <p>Post scenario validations:</p> <p>State is <i>EnergyTransferSuspended</i></p> |

StopAuthorized

| State | StopAuthorized |
|-------------------|--|
| System under test | Charging Station |
| Description | <p>This state will prepare the Charging Station, so that it is in a state where the charging session is authorized to stop. This can be done in two ways (Configurable at Test System):</p> <p>A. Using local authorization</p> <p>B. Using a RequestStopTransactionRequest</p> |

| Before (Preparations) |
|--|
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): If State is NOT <i>EnergyTransferStarted</i> then execute Reusable State <i>EnergyTransferStarted</i> |
| <p>Note: The Test System will wait a number of seconds equal to the configured <i><TransactionDuration></i>, before proceeding to the Main stage.</p> |

| Main A (Test scenario) | |
|--|---|
| Charging Station | CSMS |
| <u>Notes(s)</u> : The tool will wait for <Configured Transaction Duration> seconds | |
| <u>Manual Action</u> : Present the same idToken as used to start the transaction. | |
| 1. The Charging Station sends a TransactionEventRequest | 2. The Test System responds with a TransactionEventResponse With idTokenInfo.status is <i>Accepted</i> |
| <u>Note(s)</u> : This step is optional | |
| 3. The Charging Station sends a TransactionEventRequest | 4. The Test System responds with a TransactionEventResponse With idTokenInfo.status is <i>Accepted</i> |

| Tool validations |
|--|
| <p>* Step 1:</p> <p>Message: TransactionEventRequest</p> <ul style="list-style-type: none"> - triggerReason must be <i>StopAuthorized</i> - idToken omit OR - idToken.idToken <i><Configured valid_idtoken_idtoken></i> AND - idToken.type <i><Configured valid_idtoken_type></i> <p>* Step 3:</p> <p>Message: TransactionEventRequest</p> <ul style="list-style-type: none"> - triggerReason must be <i>ChargingStateChanged</i> - transactionInfo.chargingState must be <i>EVConnected</i> <p>If TxStopPoint contains <i>Authorized</i> or <i>PowerPathClosed</i> or <i>EnergyTransfer</i></p> <p>Then the last of the two TransactionEventRequest messages from step 1 and 3 needs to contain:</p> <ul style="list-style-type: none"> - eventType must be <i>Ended</i> - transactionInfo.stoppedReason must be <i>Local</i> or omitted <p>Else</p> <p>Then both TransactionEventRequest messages need to contain:</p> <ul style="list-style-type: none"> - eventType must be <i>Updated</i> |

| Main B (Test scenario) | |
|---|---|
| Charging Station | CSMS |
| 2. The Charging Station responds with a RequestStopTransactionResponse | 1. The Test System sends a RequestStopTransactionRequest with transactionId <transactionId provided by the Charging Station in TransactionEventRequest > |
| 3. The Charging Station sends a TransactionEventRequest | 4. The Test System responds with a TransactionEventResponse |
| <u>Note(s)</u> : This step is optional | |
| 5. The Charging Station sends a TransactionEventRequest | 6. The Test System responds with a TransactionEventResponse With idTokenInfo.status is <i>Accepted</i> |

| Tool validations |
|---|
| <p>* Step 2: Message: RequestStopTransactionResponse - status must be <i>Accepted</i></p> <p>* Step 3: Message: TransactionEventRequest - triggerReason must be <i>RemoteStop</i></p> <p>* Step 5: Message: TransactionEventRequest - triggerReason must be <i>ChargingStateChanged</i> - transactionInfo.chargingState must be <i>EVConnected</i></p> <p>If TxStopPoint contains Authorized or PowerPathClosed or EnergyTransfer Then the last of the two TransactionEventRequest messages from step 3 and 5 needs to contain: - eventType must be <i>Ended</i> - transactionInfo.stoppedReason must be <i>Remote</i></p> <p>Else Then both TransactionEventRequest messages need to contain: - eventType must be <i>Updated</i></p> |
| <p>Post scenario validations: State is <i>StopAuthorized</i></p> |

EVConnectedPostSession

| | |
|--------------------------|--|
| State | EVConnectedPostSession |
| System under test | Charging Station |
| Description | This state will prepare the Charging Station, so that energy transfer has been stopped and the transaction is NOT authorized to resume energy transfer without re-authorization. |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): If State is NOT <i>StopAuthorized</i> then execute Reusable State <i>StopAuthorized</i> |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Charging Station sends a TransactionEventRequest <u>Note(s):</u> - This step needs to be executed when the transaction has NOT been ended already. So in the case TxStopPoint contains <i>Authorized OR PowerPathClosed</i> | 2. The Test System responds with a TransactionEventResponse |
| 3. The Charging Station sends a TransactionEventRequest <u>Note(s):</u> - This step only needs to be executed when TxStopPoint contains <i>DataSigned AND</i> the transaction has NOT been ended already. So in the case TxStopPoint contains <i>Authorized OR EnergyTransfer OR PowerPathClosed</i> | 4. The Test System responds with a TransactionEventResponse |

| |
|---|
| Tool validations |
| * Step 1: Message: TransactionEventRequest - triggerReason must be <i>ChargingStateChanged</i> - transactionInfo.chargingState must be <i>EVConnected</i> * Step 3: Message: TransactionEventRequest - triggerReason must be <i>SignedDataReceived</i> |
| Post scenario validations: State is <i>EVConnectedPostSession</i> |

Deauthorized

| | |
|--------------------------|---|
| State | Deauthorized |
| System under test | Charging Station |
| Description | This reusable state will set the Charging Station to a state in which the transaction will be deauthorized. |
| Prerequisite | Reusable State <i>Authorized</i> is executed. |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): Continues executing transaction reusable states in order until the first TransactionEventRequest is received based on the configured TxStartPoint. |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Charging Station sends a TransactionEventRequest | 2. The Test System responds with a TransactionEventResponse With idTokenInfo.status is <i>Invalid</i> |
| 3. The Charging Station sends a TransactionEventRequest | 4. The Test System responds with a TransactionEventResponse |
| Note(s): - Step 3 and 4 are relevant when AuthCtrlr.StopTxOnInvalidId is <i>true</i> | |
| 6. The Charging Station sends a UnlockConnectorResponse | 5. The Test System sends a UnlockConnectorRequest Note(s): - Step 5 and 6 are executed when the connector is locked and the transaction gets deauthorized. |

| |
|--|
| Tool validations |
| * Step 1: Message: TransactionEventRequest - idToken.idToken <Configured <i>valid_idtoken_idtoken</i> > AND - idToken.type <Configured <i>valid_idtoken_type</i> > * Step 3: Message: TransactionEventRequest - triggerReason must be <i>Deauthorized</i> - transactionInfo.chargingState must be <i>EVConnected</i> or <i>SuspendedEVSE</i> If TxStopPoint contains <i>Authorized</i> or <i>PowerPathClosed</i> or <i>EnergyTransfer</i> Then : - eventType must be <i>Ended</i> - stoppedReason must be <i>DeAuthorized</i> Else: - eventType must be <i>Updated</i> |

EVDisconnected

| | |
|--------------------------|---|
| State | EVDisconnected |
| System under test | Charging Station |
| Description | This state will prepare the Charging Station, so that the EV and EVSE are disconnected, after the charging session is authorized to stop. |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State(s):

If **State** is NOT *EVConnectedPostSession* then execute **Reusable State** *EVConnectedPostSession*

Main (Test scenario)

| | |
|---|--|
| Charging Station | CSMS |
| <u>Manual Action:</u> <i>Disconnect the EV and EVSE.</i> | |
| <u>Note:</u> Steps 3 & 4 are allowed to occur before step 1. | |
| 1. The Charging Station notifies the CSMS about the status change of the connector. | 2. The Test System responds accordingly. |
| 3. The Charging Station sends a TransactionEventRequest | 4. The Test System responds with a TransactionEventResponse |
| <u>Note(s):</u> - This step needs to be executed when the transaction has NOT been ended already. So in the case TxStopPoint contains Authorized OR EnergyTransfer OR PowerPathClosed OR DataSigned | |

Tool validations

* Step 1:

Message: **StatusNotificationRequest**

- **connectorStatus** must be *Available*
- **evseld** must be *<configured evseld>*
- **connectorId** must be *<configured connectorId>*

Message: **NotifyEventRequest**

- **eventData[0].trigger** must be *Delta*
- **eventData[0].actualValue** must be *Available*
- **eventData[0].component.name** must be *Connector*
- **eventData[0].variable.name** must be *AvailabilityState*
- **eventData[0].component.evse.id** must be *<configured evseld>*
- **eventData[0].component.evse.connectorId** must be *<configured connectorId>*

* Step 3:

Message: **TransactionEventRequest**

- **triggerReason** must be *EVCommunicationLost*
- **transactionInfo.chargingState** must be *Idle*

Post scenario validations:

State is *EVDisconnected*

ParkingBayUnoccupied

| | |
|--------------------------|--|
| State | ParkingBayUnoccupied |
| System under test | Charging Station |
| Description | This state will prepare the Charging Station, so that the EV left the parking bay, after a charging session has taken place. |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): If State is NOT <i>EVDisconnected</i> then execute Reusable State <i>EVDisconnected</i> |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| Manual Action: Drive EV out of parking bay. | |
| 1. The Charging Station sends a TransactionEventRequest Note(s): - This step needs to be executed when TxStopPoint contains ParkingBayOccupancy AND the transaction has NOT been ended already. So in the case TxStopPoint contains Authorized OR EnergyTransfer OR PowerPathClosed OR DataSigned OR EVConnected. | 2. The Test System responds with a TransactionEventResponse |

| |
|---|
| Tool validations |
| * Step 1: Message: TransactionEventRequest - triggerReason must be <i>EVDeparted</i> - If the Test System is configured to stop transactions using a <i>RequestStopTransactionRequest</i> message then transactionInfo.stoppedReason must be <i>Remote</i> Else transactionInfo.stoppedReason must be <i>Local</i> - eventType must be <i>Ended</i> |
| Post scenario validations: State is <i>ParkingBayUnoccupied</i> |

StartOfflineTransaction

| | |
|--------------------------|--|
| State | StartOfflineTransaction |
| System under test | Charging Station |
| Description | This state will start a transaction while the Charging Station is offline. |
| Prerequisite | |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Test System closes the WebSocket connection AND does not accept a reconnect. | |
| <u>Manual Action:</u> Drive EV into parking bay. | |
| <u>Manual Action:</u> Present idToken. | |
| <u>Manual Action:</u> Connect the EV and EVSE. | |
| 2. The Test System accepts reconnection attempt from the Charging Station. | |

| |
|--|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

RenegotiateChargingLimits

| | |
|--------------------------|----------------------------------|
| State | RenegotiateChargingLimits |
| System under test | Charging Station |
| Description | |
| Prerequisite | |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action:</u> <i>Renegotiate EV Charging Limits</i> | |
| 1. The Charging Station sends a NotifyEVChargingNeedsRequest with evseld <Configured evseld> | 2. The Test System responds with a NotifyEVChargingNeedsResponse with status <i>Accepted</i> |
| 4. The Charging Station responds with a SetChargingProfileResponse with status <i>Accepted</i> | 3. The Test System sends a SetChargingProfileRequest with chargingProfile : .chargingProfilePurpose <i>TxProfile</i> .transactionId <Provided transactionId from before> chargingProfile.chargingSchedule[0] : .duration <i>300</i> .chargingRateUnit <Configured chargingRateUnit> <i>Note: If <Configured chargingRateUnit> is W, then the limit field will be multiplied by 1000.</i> .chargingSchedulePeriod[0].startPeriod <i>0</i> If <Configured chargingRateUnit> is W: .chargingSchedulePeriod[0].limit <i>10000</i> else: .chargingSchedulePeriod[0].limit <i>10</i> |
| 5. The Charging Station sends a NotifyEVChargingScheduleRequest with evseld <Configured evseld> | 6. The Test System responds with a NotifyEVChargingScheduleResponse with status <i>Accepted</i> |
| <u>Note:</u> Steps 5 and 6 are optional. The Charging Station will only send a NotifyEVChargingScheduleRequest when the EV returns a charging profile. | |

| Tool validations |
|--|
| <p>* Step 1:</p> <p>Message: NotifyEVChargingNeedsRequest)</p> <ul style="list-style-type: none">- evseld <Configured evseld>- if chargingNeeds.requestedEnergyTransfer is <i>DC</i>:- chargingNeeds.dcChargingParameters should not be omitted- else:- chargingNeeds.acChargingParameters should not be omitted <p>* Step 4:</p> <p>Message: SetChargingProfileResponse)</p> <ul style="list-style-type: none">- status <i>Accepted</i> <p>* Step 5:</p> <p>Message: NotifyEVChargingScheduleRequest)</p> <ul style="list-style-type: none">- evseld <Configured evseld> |
| <p>Post scenario validations:</p> <p>N/a</p> |

GetInstalledCertificates

| | |
|--------------------------|--|
| State | GetInstalledCertificates |
| System under test | Charging Station |
| Description | The hashData from installed certificates of the specified type will be retrieved from the Charging Station |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State(s):

N/a

Main (Test scenario)

| | |
|---|---|
| Charging Station | CSMS |
| 2. The Charging Station responds with a GetInstalledCertificateIdsResponse | 1. The Test System sends a GetInstalledCertificateIdsRequest With certificateType is <Specified certificateType> |

Tool validations

* Step 2:

Message: **GetInstalledCertificateIdsResponse**

- **status** must be *Accepted*
- **certificateHashDataChain** must contain an entry with following values:

Note: Order does not matter.

- **certificateHashDataChain[0].certificateType** is <Specified certificateType>
- **certificateHashDataChain[0].certificateHashData** contains <HashData from the configured certificate of the specified certificateType>

Post scenario validations:

Certificate of the specified certificateType is retrieved from the Charging Station.

RebootBeforeFirmwareInstallation

| | |
|--------------------------|--|
| State | RebootBeforeFirmwareInstallation |
| System under test | Charging Station |
| Description | The Charging Station needs to reboot before firmware <u>installation</u> . |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Charging Station sends a FirmwareStatusNotificationRequest With status <i>InstallRebooting</i> | 2. The Test System responds with a FirmwareStatusNotificationResponse |
| <u>Note:</u> The steps 3 through 8 are only executed if the bootloader is able to communicate OCPP. | |
| 3. The Charging Station sends a BootNotificationRequest | 4. The Test System responds with a BootNotificationResponse with status <i>Accepted</i> |
| 5. The Charging Station notifies the CSMS about the current state of all connectors. | 6. The Test System responds accordingly. |
| 7. The Charging Station sends a FirmwareStatusNotificationRequest With status <i>Installing</i> | 8. The Test System responds with a FirmwareStatusNotificationResponse |

| |
|--|
| Tool validations |
| * Step 1: Message FirmwareStatusNotificationRequest - status <i>InstallRebooting</i> * Step 3: Message BootNotificationRequest - reason <i>FirmwareUpdate</i> * Step 7: Message FirmwareStatusNotificationRequest - status <i>Installing</i> |
| Post scenario validations: N/a |

RebootBeforeFirmwareActivation

| | |
|--------------------------|--|
| State | RebootBeforeFirmwareActivation |
| System under test | Charging Station |
| Description | The Charging Station needs to reboot before firmware <u>activation</u> . |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Charging Station sends a FirmwareStatusNotificationRequest With status <i>InstallRebooting</i> <u>Note(s):</u> - <i>This step is optional. However it is recommended to notify the CSMS before rebooting the Charging Station to activate the new firmware.</i> | 2. The Test System responds with a FirmwareStatusNotificationResponse |
| 3. The Charging Station sends a BootNotificationRequest | 4. The Test System responds with a BootNotificationResponse with status Accepted |
| 5. The Charging Station notifies the CSMS about the current state of all connectors. | 6. The Test System responds accordingly. |

| |
|--|
| Tool validations |
| * Step 1: Message FirmwareStatusNotificationRequest - status <i>InstallRebooting</i> * Step 3: Message BootNotificationRequest - reason <i>FirmwareUpdate</i> |
| Post scenario validations: N/a |

3. Test Cases Charging Station Management System

General pre/post conditions & tool validations

General conditions/validations are overruled by testcase specific conditions/validations, unless specifically stated otherwise.

General pre conditions:

The following pre conditions apply to all test cases, unless explicitly mentioned otherwise.

- The Configuration variable **TxCtrlr.TxStartPoint** is *"EVConnected,Authorized"*
- The Configuration variable **TxCtrlr.TxStopPoint** is *"EVConnected"*
- The Configuration variable **AuthCtrlr.AuthEnabled** is *true*
- The Configuration variable **AuthCtrlr.AuthorizeRemoteStart** is *false*
- The Configuration variable **AdditionalRootCertificateCheck** is *false*
- The Configuration variable **AllowNewSessionsPendingFirmwareUpdate** is *false*
- The Configuration variable **AlignedDataSendDuringIdle** is *false*

General tool rules/validations:

- The list of ChargingSchedulePeriod elements in a chargingSchedule SHALL be ordered by increasing values of ChargingSchedulePeriod.startPeriod. This means the list is in chronological order.
- The CSMS SHALL NOT set phaseToUse in a SetChargingProfileRequest when numberPhases is other than 1.

A Security

TC_A_01_CSMS: Basic Authentication - Valid username/password combination

| | |
|--------------------------|---|
| Test case name | Basic Authentication - Valid username/password combination |
| Test case Id | TC_A_01_CSMS |
| Use case Id(s) | A00, B01 |
| Requirement(s) | A00.FR.204, B01.FR.02 |
| System under test | CSMS |
| Description | The Charging Station uses Basic authentication to authenticate itself to the CSMS, when using security profile 1 or 2. |
| Purpose | To verify whether the CSMS is able to validate the (valid) Basic authentication credentials provided by the Charging Station at the connection request. |
| Prerequisite(s) | The CSMS supports security profile 1 and/or 2 |

| |
|--|
| Before (Preparations) |
| Configuration State: The CSMS must have a password configured that equals the configured BasicAuthPassword at the Test System. |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Test System sends a HTTP upgrade request with an Authorization header, containing a username/password combination. <u>Note(s):</u> - The Authorization header is formatted as follows: AUTHORIZATION: Basic <Base64 encoded(<Configured ChargingStationId>:<Configured BasicAuthPassword>)> | 2. The CSMS validates the username/password combination AND upgrades the connection to a (secured) WebSocket connection. |
| 3. The Test System sends a BootNotificationRequest | 4. The CSMS responds with a BootNotificationResponse |
| 5. The Test System notifies the CSMS about the current state of all connectors. | 6. The CSMS responds accordingly. |

| |
|--|
| Tool validations |
| * Step 4: Message: BootNotificationResponse - status must be <i>Accepted</i> |
| Post scenario validations: N/a |

TC_A_02_CSMS: Basic Authentication - Username does not equal ChargingStationId

| | |
|--------------------------|---|
| Test case name | Basic Authentication - Username does not equal ChargingStationId |
| Test case Id | TC_A_02_CSMS |
| Use case Id(s) | A00 |
| Requirement(s) | A00.FR.204 |
| System under test | CSMS |
| Description | The Charging Station uses Basic authentication to authenticate itself to the CSMS, when using security profile 1 or 2. |
| Purpose | To verify whether the CSMS is able to validate the (invalid) Basic authentication credentials provided by the Charging Station at the connection request. |
| Prerequisite(s) | The CSMS supports security profile 1 and/or 2 |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <p>1. The Test System sends a HTTP upgrade request with an Authorization header, containing a username/password combination.</p> <p><u>Note(s):</u> - The Authorization header is formatted as follows: AUTHORIZATION: Basic <Base64 encoded(<Configured ChargingStationId> + Invalid:<Configured basicAuthPassword>)></p> | <p>2. The CSMS validates the username/password combination AND rejects the connection upgrade request.</p> |

| |
|--|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_A_03_CSMS: Basic Authentication - Invalid password

| | |
|--------------------------|---|
| Test case name | Basic Authentication - Invalid password |
| Test case Id | TC_A_03_CSMS |
| Use case Id(s) | A00 |
| Requirement(s) | A00.FR.204 |
| System under test | CSMS |
| Description | The Charging Station uses Basic authentication to authenticate itself to the CSMS, when using security profile 1 or 2. |
| Purpose | To verify whether the CSMS is able to validate the (invalid) Basic authentication credentials provided by the Charging Station at the connection request. |
| Prerequisite(s) | The CSMS supports security profile 1 and/or 2 |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Test System sends a HTTP upgrade request without an Authorization header. | 2. The CSMS rejects the connection upgrade request. |
| 3. The Test System sends a HTTP upgrade request with an Authorization header, containing a username/password combination. <u>Note(s):</u> - The Authorization header is formatted as follows: <i>AUTHORIZATION: Basic <Base64 encoded(<Configured ChargingStationId>:<randomly chosen identifierString with a sufficiently high entropy, consisting of minimum 16 and maximum 40 characters (alpha-numeric characters and the special characters allowed by identifierString)>)></i> | 4. The CSMS validates the username/password combination AND rejects the connection upgrade request. |

| |
|--|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_A_04_CSMS: TLS - server-side certificate - Valid certificate

| | |
|-------------------|---|
| Test case name | TLS - server-side certificate - Valid certificate |
| Test case Id | TC_A_04_CSMS |
| Use case Id(s) | A00 |
| Requirement(s) | A00.FR.306,A00.FR.307,A00.FR.312,A00.FR.318,A00.FR.321,A00.FR.502,A00.FR.503,A00.FR.507,A00.FR.508,A00.FR.510 |
| System under test | CSMS |
| Description | The CSMS uses a server-side certificate to identify itself to the Charging Station, when using security profile 2 or 3. |
| Purpose | To verify whether the CSMS is able to provide a valid server certificate and setup a secured WebSocket connection. |
| Prerequisite(s) | The CSMS supports security profile 2 and/or 3 |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Test System terminates the connection and initiates a TLS handshake and sends a Client Hello to the CSMS. | 2. The CSMS responds with a Server Hello With the <Configured server certificate> |
| 3. The Test System performs the following actions: Send client certificate Client Key Exchange Certificate verify Change Cipher Spec Finished <u>Note(s):</u> - The client certificate is only sent when the CSMS uses security profile 3. | 4. The CSMS performs the following actions: Change Cipher Spec Finished |
| 5. The Test System sends a HTTP upgrade request to the CSMS <u>Note(s):</u> - The HTTP request only contains a username/password combination when the CSMS uses security profile 2. | 6. The CSMS upgrades the connection to a (secured) WebSocket connection. |
| 7. The Test System sends a BootNotificationRequest with reason <i>PowerUp</i> chargingStation.model <Configured model> chargingStation.vendorName <Configured vendorName> | 8. The CSMS responds with a BootNotificationResponse |

| Main (Test scenario) | |
|---|--|
| <p>9. The Test System notifies the CSMS about the current state of all connectors.</p> <p>Message: StatusNotificationRequest</p> <ul style="list-style-type: none"> - connectorStatus <i>Available</i> <p>Message: NotifyEventRequest</p> <ul style="list-style-type: none"> - trigger <i>Delta</i> - actualValue <i>"Available"</i> - component.name <i>"Connector"</i> - variable.name <i>"AvailabilityState"</i> | <p>10. The CSMS responds accordingly.</p> |
| Tool validations | |
| <p>* Step 3:</p> <p>The Test System validates the following before finishing the TLS handshake:</p> <ul style="list-style-type: none"> - The CSMS must use TLS version 1.2 or above <p>At least the following set of cipher suites must be supported:</p> <p>TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256</p> <p>AND</p> <p>TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384</p> <p>AND</p> <p>TLS_RSA_WITH_AES_128_GCM_SHA256</p> <p>AND</p> <p>TLS_RSA_WITH_AES_256_GCM_SHA384</p> <ul style="list-style-type: none"> - When using RSA or DSA the key must be at least 2048 bits long. <p>and when using elliptic curve cryptography the key must be at least 224 bits long.</p> <ul style="list-style-type: none"> - The received server side certificate must be transmitted in the X.509 format encoded in Privacy-Enhanced Mail (PEM) format. - The certificate must include a serial number. - The subject field of the certificate must contain a commonName RDN which consists of the FQDN of the endpoint of the server. <p><i>NOTE: If one of the above validations fails, the Test System can still proceed with the next steps of the testcase (if it is able to), but the testcase will FAIL and the Test System reports why it failed.</i></p> <p>* Step 8:</p> <p>Message: BootNotificationResponse</p> <p>with status <i>Accepted</i></p> | |
| <p>Post scenario validations:</p> <p>N/a</p> | |

TC_A_06_CSMS: TLS - server-side certificate - TLS version too low

| | |
|-------------------|---|
| Test case name | TLS - server-side certificate - TLS version too low |
| Test case Id | TC_A_06_CSMS |
| Use case Id(s) | A00 |
| Requirement(s) | A00.FR.314,A00.FR.315,A00.FR.409,A00.FR.416,A00.FR.417,A00.FR.418 |
| System under test | CSMS |
| Description | The CSMS uses a server-side certificate to identify itself to the Charging Station, when using security profile 2 or 3. |
| Purpose | To verify whether the CSMS is able to terminate the connection when it notices the used TLS version is lower than 1.2. |
| Prerequisite(s) | The CSMS supports security profile 2 and/or 3 |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Test System terminates the connection and initiates a TLS handshake with a TLS version lower than 1.2 and sends a Client Hello to the CSMS. | 2. The CSMS notices that the TLS version is lower than 1.2 and terminates the connection. |
| 3. The Test System initiates a TLS handshake with TLS version 1.2 or higher and sends a Client Hello to the CSMS. | 4. The CSMS responds with a Server Hello With the <Configured server certificate> |
| 5. The Test System performs the following actions: Send client certificate Client Key Exchange Certificate verify Change Cipher Spec Finished <u>Note(s):</u> - The client certificate is only sent when the CSMS uses security profile 3. | 6. The CSMS performs the following actions: Change Cipher Spec Finished |
| 7. The Test System sends a HTTP upgrade request to the CSMS <u>Note(s):</u> - The HTTP request only contains a username/password combination when the CSMS uses security profile 2. | 8. The CSMS upgrades the connection to a (secured) WebSocket connection. |
| 9. The Test System sends a BootNotificationRequest with reason <i>PowerUp</i> chargingStation.model <Configured model> chargingStation.vendorName <Configured vendorName> | 10. The CSMS responds with a BootNotificationResponse |

| Main (Test scenario) | |
|--|--|
| <p>11. The Test System notifies the CSMS about the current state of all connectors.</p> <p>Message: StatusNotificationRequest</p> <ul style="list-style-type: none">- connectorStatus <i>Available</i> <p>Message: NotifyEventRequest</p> <ul style="list-style-type: none">- trigger <i>Delta</i>- actualValue <i>"Available"</i>- component.name <i>"Connector"</i>- variable.name <i>"AvailabilityState"</i> | <p>12. The CSMS responds accordingly.</p> |
| Tool validations | |
| <p>* Step 10:</p> <p>Message: BootNotificationResponse</p> <ul style="list-style-type: none">- status <i>Accepted</i> | |
| <p>Post scenario validations:</p> <p>N/a</p> | |

TC_A_07_CSMS: TLS - Client-side certificate - valid certificate

| | |
|-------------------|---|
| Test case name | TLS - Client-side certificate - valid certificate |
| Test case Id | TC_A_07_CSMS |
| Use case Id(s) | A00 |
| Requirement(s) | A00.FR.409,A00.FR.410,A00.FR.415,A00.FR.416,A00.FR.421 |
| System under test | CSMS |
| Description | The Charging Station uses a client-side certificate to identify itself to the CSMS, when using security profile 3. |
| Purpose | To verify whether the CSMS is able to receive a client certificate provided by a Charging Station and setup a secured WebSocket connection. |
| Prerequisite(s) | The CSMS supports security profile 3 |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Test System terminates the connection and initiates a TLS handshake and sends a Client Hello to the CSMS. | 2. The CSMS responds with a Server Hello With the <Configured server certificate> |
| 3. The Test System performs the following actions: Send <Configured client certificate> Client Key Exchange Certificate verify Change Cipher Spec Finished | 4. The CSMS performs the following actions: Change Cipher Spec Finished |
| 5. The Test System sends a HTTP upgrade request to the CSMS | 6. The CSMS upgrades the connection to a (secured) WebSocket connection. |
| 7. The Test System sends a BootNotificationRequest with reason <i>PowerUp</i> chargingStation.model <Configured model> chargingStation.vendorName <Configured vendorName> | 8. The CSMS responds with a BootNotificationResponse |
| 9. The Test System notifies the CSMS about the current state of all connectors. Message: StatusNotificationRequest - connectorStatus <i>Available</i> Message: NotifyEventRequest - trigger <i>Delta</i> - actualValue <i>"Available"</i> - component.name <i>"Connector"</i> - variable.name <i>"AvailabilityState"</i> | 10. The CSMS responds accordingly. |

| Tool validations |
|---|
| <p>* Step 3:</p> <p>The Test System validates the following before finishing the TLS handshake:</p> <ul style="list-style-type: none">- The CSMS must use TLS version 1.2 or above <p>At least the following set of cipher suites must be supported:</p> <p>TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256</p> <p>AND</p> <p>TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384</p> <p>AND</p> <p>TLS_RSA_WITH_AES_128_GCM_SHA256</p> <p>AND</p> <p>TLS_RSA_WITH_AES_256_GCM_SHA384</p> <p>* Step 8:</p> <p>Message: BootNotificationResponse</p> <p>with status <i>Accepted</i></p> |
| <p>Post scenario validations:</p> <p>N/a</p> |

TC_A_08_CSMS: TLS - Client-side certificate - Invalid certificate

| | |
|-------------------|---|
| Test case name | TLS - Client-side certificate - Invalid certificate |
| Test case Id | TC_A_08_CSMS |
| Use case Id(s) | A00 |
| Requirement(s) | A00.FR.405,A00.FR.407,A00.FR.409,A00.FR.410 |
| System under test | CSMS |
| Description | The Charging Station uses a client-side certificate to identify itself to the CSMS, when using security profile 3. |
| Purpose | To verify whether the CSMS is able to terminate the connection when the received client certificate is invalid. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The CSMS supports security profile 3 - This testcase can be executed multiple times, using different kinds of invalid certificates: Unknown certificate expired certificate certificate with commonName that does not equal the serial number of the Charging Station. |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Test System initiates a TLS handshake and sends a Client Hello to the CSMS. | 2. The CSMS responds with a Server Hello With a server certificate |
| 3. The Test System performs the following actions: Send <Configured invalid client certificate> Client Key Exchange Certificate verify Change Cipher Spec Finished | 4. The CSMS deems the client certificate invalid and terminates the connection. |
| 5. The Test System initiates a TLS handshake and sends a Client Hello to the CSMS. | 6. The CSMS responds with a Server Hello With a server certificate |
| 7. The Test System performs the following actions: Send <Configured client certificate> Client Key Exchange Certificate verify Change Cipher Spec Finished | 8. The CSMS performs the following actions: Change Cipher Spec Finished |
| 9. The Test System sends a HTTP upgrade request to the CSMS | 10. The CSMS upgrades the connection to a (secured) WebSocket connection. |
| 11. The Test System sends a BootNotificationRequest with reason <i>PowerUp</i> chargingStation.model <Configured model> chargingStation.vendorName <Configured vendorName> | 12. The CSMS responds with a BootNotificationResponse |

| Main (Test scenario) | |
|--|--|
| <p>13. The Test System notifies the CSMS about the current state of all connectors.</p> <p>Message: StatusNotificationRequest</p> <ul style="list-style-type: none">- connectorStatus <i>Available</i> <p>Message: NotifyEventRequest</p> <ul style="list-style-type: none">- trigger <i>Delta</i>- actualValue <i>"Available"</i>- component.name <i>"Connector"</i>- variable.name <i>"AvailabilityState"</i> | <p>14. The CSMS responds accordingly.</p> |

| Tool validations |
|--|
| <p>* Step 12:</p> <p>Message: BootNotificationResponse with status <i>Accepted</i></p> |
| <p>Post scenario validations: N/a</p> |

TC_A_09_CSMS: Update Charging Station Password for HTTP Basic Authentication - Accepted

| | |
|-------------------|--|
| Test case name | Update Charging Station Password for HTTP Basic Authentication - Accepted |
| Test case Id | TC_A_09_CSMS |
| Use case Id(s) | A01 |
| Requirement(s) | A01.FR.02, A01.FR.03 |
| System under test | CSMS |
| Description | This test case defines how to use the BasicAuthPassword, the password used to authenticate Charging Stations in security profile 1 (Basic Authentication) and security profile 2 (TLS with Basic Authentication) |
| Purpose | To verify if the CSMS is able to successfully set the new BasicAuthPassword and only accepts the new credentials as described at the OCPP specification. |
| Prerequisite(s) | The CSMS supports security profile 1 and/or 2 |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a SetVariablesResponse with status Accepted | 1. The CSMS sends a SetVariablesRequest with: setVariableData[1] : - variable.name = "BasicAuthPassword" - component.name = "SecurityCtrlr" - attributeValue = "<NewPassword>" |
| 3. The Test System sends a HTTP upgrade request with an Authorization header, containing a username/password combination (with the new <i>BasicAuthPassword</i>). <u>Note(s)</u> : - The Authorization header is formatted as follows: AUTHORIZATION: Basic <Base64 encoded(<Configured ChargingStationId>:<NEW BasicAuthPassword>)> | 4. The CSMS validates the username/password combination AND upgrades the connection to a (secured) WebSocket connection. |
| 5. The Test System sends a BootNotificationRequest | 6. The CSMS responds with a BootNotificationResponse |
| 7. The Test System notifies the CSMS about the current state of all connectors. | 8. The CSMS responds accordingly. |

| |
|--|
| Tool validations |
| * Step 1: Message: SetVariableRequest - variable.name = "BasicAuthPassword" - component.name = "SecurityCtrlr" |
| * Step 6: Message: BootNotificationResponse - status must be <i>Accepted</i> |
| Post scenario validations: N/a |

TC_A_10_CSMS: Update Charging Station Password for HTTP Basic Authentication - Rejected

| | |
|-------------------|--|
| Test case name | Update Charging Station Password for HTTP Basic Authentication - Rejected |
| Test case Id | TC_A_10_CSMS |
| Use case Id(s) | A01 |
| Requirement(s) | A01.FR.02, A01.FR.04, A01.FR.05 |
| System under test | CSMS |
| Description | This test case defines how to use the BasicAuthPassword, the password used to authenticate Charging Stations in security profile 1 (Basic Authentication) and security profile 2 (TLS with Basic Authentication) |
| Purpose | To verify if the CSMS keeps accepting the old credentials and keeps communication when the new BasicAuthPassword is rejected as described at the OCPP specification. |
| Prerequisite(s) | The CSMS supports security profile 1 and/or 2 |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a SetVariablesResponse with status <i>Rejected</i> | <p>1. The CSMS sends a SetVariablesRequest with:</p> <p>setVariableData[1]:</p> <ul style="list-style-type: none"> - variable.name = "BasicAuthPassword" - component.name = "SecurityCtrlr" - attributeValue = "<NewPassword>" |
| | <p>3. The Test System sends a HTTP upgrade request with an Authorization header, containing a username/password combination (with the old <i>BasicAuthPassword</i>).</p> <p><u>Note(s):</u></p> <ul style="list-style-type: none"> - The Authorization header is formatted as follows: AUTHORIZATION: Basic <Base64 encoded(<Configured ChargingStationId>:<OLD Configured BasicAuthPassword>)> |
| 4. The CSMS validates the username/password combination AND upgrades the connection to a (secured) WebSocket connection. | |
| 5. The Test System sends a BootNotificationRequest | 6. The CSMS responds with a BootNotificationResponse |
| | 7. The Test System notifies the CSMS about the current state of all connectors. |
| 8. The CSMS responds accordingly. | |

| Tool validations |
|--|
| <p>* Step 1:</p> <p>Message: SetVariableRequest</p> <ul style="list-style-type: none">- variable.name = <i>"BasicAuthPassword"</i>- component.name = <i>"SecurityCtrlr"</i> <p>* Step 6:</p> <p>Message: BootNotificationResponse</p> <ul style="list-style-type: none">- status must be <i>Accepted</i> |
| <p>Post scenario validations:</p> <p>N/a</p> |

TC_A_11_CSMS: Update Charging Station Certificate by request of CSMS - Success - Charging Station Certificate

| | |
|--------------------------|--|
| Test case name | Update Charging Station Certificate by request of CSMS - Success - Charging Station Certificate |
| Test case Id | TC_A_11_CSMS |
| Use case Id(s) | A02 & F06 |
| Requirement(s) | A02.FR.11, A02.FR.14, F06.FR.01 |
| System under test | CSMS |
| Description | The CSMS is able to request the Charging Station to update its charging station certificate using the TriggerMessageRequest message. |
| Purpose | To verify if the CSMS is able to request the Charging Station to update its Charging Station Certificate. |
| Prerequisite(s) | The CSMS supports security profile 3 |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State RenewChargingStationCertificate | |
| 2. The Test System disconnects its current connection and reconnects to the CSMS with the new certificate. | 3. The CSMS accepts the incoming connection request using the new certificate. |

| |
|--|
| Tool validations |
| N/a |
| Post scenario validations: The Test System and the CSMS are connected. |

TC_A_12_CSMS: Update Charging Station Certificate by request of CSMS - Success - V2G Certificate

| | |
|-------------------|--|
| Test case name | Update Charging Station Certificate by request of CSMS - Success - V2G Certificate |
| Test case Id | TC_A_12_CSMS |
| Use case Id(s) | A02 & F06 |
| Requirement(s) | A02.FR.11, F06.FR.01 |
| System under test | CSMS |
| Description | The CSMS is able to request the Charging Station to update its charging station certificate using the TriggerMessageRequest message. |
| Purpose | To verify if the CSMS is able to request the Charging Station to update its V2G Certificate. |
| Prerequisite(s) | The CSMS supports ISO 15118. |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a TriggerMessageResponse With status <i>Accepted</i> | 1. The CSMS sends a TriggerMessageRequest |
| 3 The Test System sends a SignCertificateRequest With csr <i>Generated CSR based on:</i> - <Configured Country> - <Configured Organization> - <Configured OrganizationalUnit> certificateType <i>V2GCertificate</i> | 4. The CSMS responds with a SignCertificateResponse |
| 6. The Test System responds with a CertificateSignedResponse With status <i>Accepted</i> | 5. The CSMS sends a CertificateSignedRequest |

| Tool validations |
|--|
| <p>* Step 1:</p> <p>Message: TriggerMessageRequest</p> <ul style="list-style-type: none">- requestedMessage <i>SignV2GCertificate</i> <p>* Step 4:</p> <p>Message: SignCertificateResponse</p> <ul style="list-style-type: none">- status <i>Accepted</i> <p>* Step 5:</p> <p>Message: CertificateSignedRequest</p> <ul style="list-style-type: none">- certificateChain <i><Certificate generated from the received CSR from step 3 and signed by the V2G Root or SubCA certificate from the configured V2G certificate chain></i> <p><i>NOTE: The Test System will validate the certificate, but if the following validation fail, the testcase will NOT FAIL, because generating the certificate is probably not be done by the CSMS.</i></p> <ul style="list-style-type: none">- The key must be at least 224 bits long.- The received certificate must be transmitted in the X.509 format encoded in Privacy-Enhanced Mail (PEM) format. |
| <p>Post scenario validations:</p> <p>N/a</p> |

TC_A_14_CSMS: Update Charging Station Certificate by request of CSMS - Invalid certificate

| | |
|-------------------|--|
| Test case name | Update Charging Station Certificate by request of CSMS - Invalid certificate |
| Test case Id | TC_A_14_CSMS |
| Use case Id(s) | A02 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | The CSMS is able to request the Charging Station to update its charging station certificate using the TriggerMessageRequest message. |
| Purpose | To verify if the CSMS is able to handle a Charging Station rejecting the new Charging Station certificate. |
| Prerequisite(s) | The CSMS supports security profile 3 |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a TriggerMessageResponse With status <i>Accepted</i> | 1. The CSMS sends a TriggerMessageRequest |
| 3 The Test System sends a SignCertificateRequest With csr <i><Configured CSR></i> certificateType <i>ChargingStationCertificate</i> | 4. The CSMS responds with a SignCertificateResponse |
| 6. The Test System responds with a CertificateSignedResponse With status <i>Rejected</i> | 5. The CSMS sends a CertificateSignedRequest |
| 7. The Test System sends a SecurityEventNotificationRequest with type = <i>InvalidChargingStationCertificate</i> | 8. The CSMS responds with a SecurityEventNotificationResponse |

| |
|---|
| Tool validations |
| * Step 1: Message: TriggerMessageRequest - requestedMessage <i>SignChargingStationCertificate</i> |
| * Step 4: Message: SignCertificateResponse - status <i>Accepted</i> |
| Post scenario validations: N/a |

TC_A_19_CSMS: Upgrade Charging Station Security Profile - Accepted

| | |
|-------------------|---|
| Test case name | Upgrade Charging Station Security Profile - Accepted |
| Test case Id | TC_A_19_CSMS |
| Use case Id(s) | A05 |
| Requirement(s) | A05.FR.04, A05.FR.07 |
| System under test | CSMS |
| Description | The CSMS updates the connection details on the Charging Station, to increase the security profile level. |
| Purpose | To verify if the CSMS is able to set a new network connection profile at one of the by the Charging Station defined configuration slots with a higher security profile than currently configured. |
| Prerequisite(s) | <ul style="list-style-type: none"> - Security profile must be set to 1 or 2. - If Security profile is set to 1, then a trusted certificate must be installed. |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: If configured <Security profile> is 2, then RenewChargingStationCertificate The Test System uses this certificate during the TLS handshake when connecting with security profile 3. |
| Reusable State(s): N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action:</u> Request the CSMS to set a new NetworkConnectionProfile with a security profile level one higher than currently configured | |
| 2. The Test System responds with a SetNetworkProfileResponse With status Accepted | 1. The CSMS sends a SetNetworkProfileRequest |
| <u>Manual Action:</u> Request the CSMS to change the NetworkConfigurationPriority to one that contains the configurationSlot of the new NetworkConnectionProfile from step 1 | |
| 4. The Test System responds with a SetVariablesResponse with status Accepted | 3. The CSMS sends a SetVariablesRequest |
| <u>Manual Action:</u> Request the CSMS to reboot the Charging Station | |
| 6. The Test System responds with a ResetResponse with status Accepted | 5. The CSMS sends a ResetRequest |
| 7. The Test System reconnects to the CSMS using the new NetworkProfile, containing the upgraded security profile <Configured securityProfile + 1>. | 8. The CSMS accepts the connection attempt. |
| 9. Execute Reusable State Booted | |
| 10. The Test System reconnects to the CSMS using the original NetworkProfile, containing the lower security profile. | 11. The CSMS shall not accept the connection attempt. |
| <u>Note(s):</u> - This is done to ensure that the CSMS does not accept a connection using the lower security profile anymore. | |

| Main (Test scenario) | |
|--|--|
| <p>12. The Test System reconnects to the CSMS using the new NetworkProfile, containing the upgraded security profile <Configured securityProfile + 1>.</p> <p><u>Note(s):</u> - This is done to restore the connection before ending the testcase.</p> | <p>13. The CSMS accepts the connection attempt.</p> |

| Tool validations |
|---|
| <p>* Step 1: Message SetNetworkProfileRequest - connectionData.messageTimeout <Configured messageTimeout> - connectionData.ocppInterface <Configured ocppInterface> - connectionData.ocppTransport JSON - connectionData.ocppVersion OCPP20 - connectionData.securityProfile <Configured securityProfile + 1></p> <p>* Step 3: Message SetVariablesRequest setVariableData: - variable.name = "NetworkConfigurationPriority" - component.name = "OCPPCommCtrlr" - attributeValue = <contains configurationSlot provided at step 1></p> <p>* Step 11: When upgrading a Charging Station to a higher security profile, a CSMS has several options regarding which endpoint to use. This affects the way the CSMS is able to detect it needs to reject the incoming connection attempt.</p> <p>In case of having upgraded from security profile 2 to 3, but there is an incoming connection attempt using security profile 2: When the same endpoint is used, then it depends on the CSMS endpoint configuration. - When the CSMS does a full switch and only allows TLS handshakes when a client certificate is provided, then the TLS handshake is rejected. - When the CSMS only requires this Charging Station to use a client certificate, then it accepts the TLS handshake (because it will be unable to detect which Charging Station is connecting) and it rejects the HTTP request to establish the WebSocket connection.</p> <p>When a different port or a whole different endpoint is used for the upgrade, then on the original endpoint the CSMS accepts the TLS handshake and it rejects the HTTP request to establish the WebSocket connection (because this Charging Station is not allowed to connect with security profile 2 anymore).</p> <p>In case of security profile 1, the case is always the same. The CSMS shall always reject the HTTP request to establish the WebSocket connection, because TLS is required for this Charging Station.</p> <p>Post scenario validations: The Test System and the CSMS are connected.</p> |

B Provisioning**TC_B_01_CSMS: Cold Boot Charging Station - Accepted**

| | |
|--------------------------|---|
| Test case name | Cold Boot Charging Station - Accepted |
| Test case Id | TC_B_01_CSMS |
| Use case Id(s) | B01 |
| Requirement(s) | B01.FR.02 |
| System under test | CSMS |
| Description | The booting mechanism allows a Charging Station to provide some general information about the Charging Station to the CSMS on startup AND it allows the Charging Station to request whether it is allowed to start sending other OCPP messages. |
| Purpose | To verify whether the CSMS is able to accept the communications of a registered Charging Station. |
| Prerequisite(s) | N/a |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>Booted</i> | |

| |
|--|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_B_02_CSMS: Cold Boot Charging Station - Pending

| | |
|-------------------|--|
| Test case name | Cold Boot Charging Station - Pending |
| Test case Id | TC_B_02_CSMS |
| Use case Id(s) | B02 |
| Requirement(s) | B02.FR.01, B02.FR.06 |
| System under test | CSMS |
| Description | The booting mechanism allows a Charging Station to provide some general information about the Charging Station to the CSMS on startup AND it allows the Charging Station to request whether it is allowed to start sending other OCPP messages. The CSMS may respond to the BootNotificationRequest with status <i>Pending</i> . The <i>Pending</i> status can indicate that the CSMS wants to retrieve or set certain information on the Charging Station before it will accept the Charging Station. |
| Purpose | To verify whether the CSMS is able to accept the communications of a registered Charging Station. |
| Prerequisite(s) | The CSMS is configured to first respond to a BootNotificationRequest with status Pending . |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Test System sends a BootNotificationRequest with reason <i>PowerUp</i> chargingStation.model <Configured model> chargingStation.vendorName <Configured vendorName> | 2. The CSMS responds with a BootNotificationResponse |
| <u>Note(s):</u> - If the interval in the BootNotificationResponse equals 0, the Test System will wait <Configured heartbeatInterval> seconds, before sending another BootNotificationRequest. - If the interval in the BootNotificationResponse > 0, the Test System will wait <Interval provided at the BootNotificationResponse> seconds, before sending another BootNotificationRequest. - During this interval, the CSMS may send messages to retrieve information from the Charging Station (as described in use cases B06, B07, B08) or change its configuration by SetVariablesRequest (as described in use case B05). The Test System will respond to these messages. | |
| 3. The Test System sends a BootNotificationRequest with reason <i>PowerUp</i> chargingStation.model <Configured model> chargingStation.vendorName <Configured vendorName> | 4. The CSMS responds with a BootNotificationResponse |
| 5. The Test System notifies the CSMS about the current state of all connectors. Message: StatusNotificationRequest with connectorStatus <i>Available</i> Message: NotifyEventRequest with trigger <i>Delta</i> actualValue "Available" component.name "Connector" variable.name "AvailabilityState" | 6. The CSMS responds accordingly. |

| Tool validations |
|--|
| <p>* Step 2:</p> <p>Message: BootNotificationResponse</p> <p>- status <i>Pending</i></p> <p>* Step 3:</p> <p>Message: BootNotificationResponse</p> <p>- status <i>Accepted</i></p> |
| <p>Post scenario validations:</p> <p>N/a</p> |

TC_B_30_CSMS: Cold Boot Charging Station - Pending/Rejected - SecurityError

| | |
|-------------------|---|
| Test case name | Cold Boot Charging Station - Pending/Rejected - SecurityError |
| Test case Id | TC_B_30_CSMS |
| Use case Id(s) | B02/B03 |
| Requirement(s) | B02.FR.09, B03.FR.07 |
| System under test | CSMS |
| Description | The booting mechanism allows a Charging Station to provide some general information about the Charging Station to the CSMS on startup AND it allows the Charging Station to request whether it is allowed to start sending other OCPP messages. The CSMS may respond to the BootNotificationRequest with status <i>Pending</i> or <i>Rejected</i> . During this state, the Charging Station is not allowed to send RPC Framework: CALL message that is NOT a BootNotificationRequest or in case of status <i>Pending</i> , a message triggered by one of the following messages: TriggerMessageRequest, GetBaseReportRequest, GetReportRequest. |
| Purpose | To verify whether the CSMS is able to handle unauthorized messages from the Charging Station by responding with a SecurityError. |
| Prerequisite(s) | The CSMS is configured to first respond to a BootNotificationRequest with status <i>Pending</i> or <i>Rejected</i> . |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State(s):

N/a

Main (Test scenario)

| Charging Station | CSMS |
|---|--|
| 1. The Test System sends a BootNotificationRequest with reason <i>PowerUp</i> chargingStation.model <Configured model> chargingStation.vendorName <Configured vendorName> | 2. The CSMS responds with a BootNotificationResponse |
| 3. The Test System notifies the CSMS about the current state of all connectors. Message: StatusNotificationRequest with connectorStatus <i>Available</i> Message: NotifyEventRequest with trigger <i>Delta</i> actualValue "Available" component.name "Connector" variable.name "AvailabilityState" | 4. The CSMS responds with RPC Framework: CALLERROR: SecurityError. |

Tool validations

* Step 2:

Message: **BootNotificationResponse**- **status** *Pending* OR *Rejected*

Post scenario validations:

N/a

TC_B_31_CSMS: Cold Boot Charging Station - Pending/Rejected - TriggerMessage

| | |
|-------------------|---|
| Test case name | Cold Boot Charging Station - Pending/Rejected - TriggerMessage |
| Test case Id | TC_B_31_CSMS |
| Use case Id(s) | B02, F06 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | The booting mechanism allows a Charging Station to provide some general information about the Charging Station to the CSMS on startup AND it allows the Charging Station to request whether it is allowed to start sending other OCPP messages. |
| Purpose | To verify whether the CSMS is able to send a TriggerMessageRequest to trigger a BootNotificationRequest, before the interval expired. |
| Prerequisite(s) | The CSMS is configured to first respond to a BootNotificationRequest with status <i>Pending</i> or <i>Rejected</i> . |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Test System sends a BootNotificationRequest with reason <i>PowerUp</i> chargingStation.model <Configured model> chargingStation.vendorName <Configured vendorName> | 2. The CSMS responds with a BootNotificationResponse |
| 4. The Test System responds with a TriggerMessageResponse with status <i>Accepted</i> | 3. The CSMS sends a TriggerMessageRequest |
| 5. The Test System sends a BootNotificationRequest with reason <i>Triggered</i> chargingStation.model <Configured model> chargingStation.vendorName <Configured vendorName> | 6. The CSMS responds with a BootNotificationResponse |
| 7. The Test System notifies the CSMS about the current state of all connectors. Message: StatusNotificationRequest with connectorStatus <i>Available</i> Message: NotifyEventRequest with trigger <i>Delta</i> actualValue "Available" component.name "Connector" variable.name "AvailabilityState" | 8. The CSMS responds accordingly. |

| Tool validations |
|---|
| <div>* Step 2: Message: BootNotificationResponse - status <i>Pending OR Rejected</i></div> <div>* Step 3: Message: TriggerMessageRequest - requestedMessage <i>BootNotification</i></div> <div>* Step 6: Message: BootNotificationResponse - status <i>Accepted</i></div> |
| Post scenario validations: N/a |

TC_B_06_CSMS: Get Variables - single value

| | |
|-------------------|---|
| Test case name | Get Variables - single value |
| Test case Id | TC_B_06_CSMS |
| Use case Id(s) | B06 |
| Requirement(s) | B06.FR.01, B06.FR.02, B06.FR.03, B06.FR.04, B06.FR.10, B06.FR.11 |
| System under test | CSMS |
| Description | Get the value of two of the required variables of OCPPCommCtrlr |
| Purpose | To test getting single value using GetVariablesRequest for one of the mandatory component/variable combinations that must exist in the DM implementation. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. Test System responds with: GetVariablesResponse | 1. <i>Manually request CSMS to get data for:</i> - OCPPCommCtrlr.OfflineThreshold |

| |
|---|
| Tool validations |
| <p>* Step 1:</p> <p>Message: GetVariablesRequest with (in arbitrary order)</p> <p>getVariableData[0]:</p> <ul style="list-style-type: none"> - attributeType is at least absent or attributeType = <i>Actual</i>, but <i>Target</i>, <i>MinSet</i>, and <i>MaxSet</i> are also allowed - variable.name = "OfflineThreshold" - component.name = "OCPPCommCtrlr" |
| <p>Post scenario validations:</p> <p>Manually validate that CSMS has correctly read the requested variables.</p> |

TC_B_07_CSMS: Get Variables - multiple values

| | |
|-------------------|--|
| Test case name | Get Variables - multiple values |
| Test case Id | TC_B_07_CSMS |
| Use case Id(s) | B06 |
| Requirement(s) | B06.FR.01, B06.FR.02, B06.FR.03 |
| System under test | CSMS |
| Description | Get the value of two of the required variables of OCPPCommCtrlr |
| Purpose | To test getting multiple values using GetVariablesRequest for one of the mandatory component/variable combinations that must exist in the DM implementation. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. Test System responds with: GetVariablesResponse | 1. Manually request CSMS to get data for: - OCPPCommCtrlr.OfflineThreshold - AuthCtrlr.AuthorizeRemoteStart |

| |
|---|
| Tool validations |
| <p>* Step 1:</p> <p>Message: GetVariablesRequest with (in arbitrary order)</p> <p>getVariableData[0]:</p> <ul style="list-style-type: none"> - attributeType is at least absent or attributeType = <i>Actual</i>, but <i>Target</i>, <i>MinSet</i>, and <i>MaxSet</i> are also allowed - variable.name = "OfflineThreshold" - component.name = "OCPPCommCtrlr" <p>getVariableData[1]:</p> <ul style="list-style-type: none"> - attributeType is at least absent or attributeType = <i>Actual</i>, but <i>Target</i>, <i>MinSet</i>, and <i>MaxSet</i> are also allowed - variable.name = "AuthorizeRemoteStart" - component.name = "AuthCtrlr" |
| Post scenario validations: Manually validate that CSMS has correctly read the requested variables. |

TC_B_08_CSMS: Get Variables - limit to maximum number of values

| | |
|-------------------|---|
| Test case name | Get Variables - limit to maximum number of values |
| Test case Id | TC_B_08_CSMS |
| Use case Id(s) | B06 |
| Requirement(s) | B06.FR.05 |
| System under test | CSMS |
| Description | Do not request more variables than supported by <code>MaxItemsPerMessageGetVariables</code> . |
| Purpose | To test that CSMS does not request more variables than the Charging Station reported to support in the variable <code>MaxItemsPerMessageGetVariables</code> . |
| Prerequisite(s) | N/a |

Before (Preparations)

| |
|--|
| Configuration State: Configure (using <code>getVariablesRequest</code>) <code>Component.Variable.Instance DeviceDataCtrlr.ItemsPerMessage.GetVariables</code> at value 4. |
| Memory State: N/a |
| Reusable State(s): N/a |

Main (Test scenario)

| Charging Station | CSMS |
|--|--|
| 2. Test System responds with GetVariablesResponse with with a list of 4 GetVariableResultType items and a GetVariableResponse with 1 GetVariableResultType item. | 1. Manually request CSMS for 5 variables: - <code>DeviceDataCtrlr.ItemsPerMessage[GetReport]</code> - <code>DeviceDataCtrlr.ItemsPerMessage[GetVariables]</code> - <code>DeviceDataCtrlr.BytesPerMessage[GetReport]</code> - <code>DeviceDataCtrlr.BytesPerMessage[GetVariables]</code> - <code>AuthCtrlr.AuthorizeRemoteStart</code> |

Tool validations

| |
|---|
| * Step 1: Message: GetVariablesRequest for 4 variables and a GetVariablesRequest for 1 variable (in arbitrary order): for <code>component.name = "DeviceDataCtrlr"</code> - <code>variable.name = "ItemsPerMessage"</code> with <code>variable.instance = "GetReport"</code> - <code>variable.name = "ItemsPerMessage"</code> with <code>variable.instance = "GetVariables"</code> - <code>variable.name = "BytesPerMessage"</code> with <code>variable.instance = "GetReport"</code> - <code>variable.name = "BytesPerMessage"</code> with <code>variable.instance = "GetVariables"</code> and for <code>component.name = "AuthCtrlr"</code> - <code>variable.name = "AuthorizeRemoteStart"</code> |
| Post scenario validations: Test System validates that not more than <code>ItemsPerMessageGetVariables</code> elements are requested in one GetVariablesRequest message by CSMS. |

TC_B_09_CSMS: Set Variables - single value

| | |
|-------------------|---|
| Test case name | Set Variables - single value |
| Test case Id | TC_B_09_CSMS |
| Use case Id(s) | B05 |
| Requirement(s) | B05.FR.01, B05.FR.02, B05.FR.03, B05.FR.10, B05.FR.12 |
| System under test | CSMS |
| Description | Set the value of one of the required variables of OCPPCommCtrlr |
| Purpose | To test setting a single value using SetVariablesRequest for one of the mandatory component/variable combinations that must exist in the DM implementation. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. Test System responds with: SetVariablesResponse | 1. <i>Manually request CSMS to set data for:</i> - OCPPCommCtrlr.OfflineThreshold |

| |
|--|
| Tool validations |
| <p>* Step 1:</p> <p>Message: SetVariablesRequest with (in arbitraty order):</p> <p>setVariableData[1]:</p> <ul style="list-style-type: none"> - variable.name = "OfflineThreshold" - component.name = "OCPPCommCtrlr" - attributeValue = "123" - attributeType is absent or attributeType = <i>Actual</i> |
| <p>Post scenario validations:</p> <p>Manually validate that CSMS has correctly set the requested variables.</p> |

TC_B_10_CSMS: Set Variables - multiple values

| | |
|-------------------|--|
| Test case name | Set Variables - multiple values |
| Test case Id | TC_B_10_CSMS |
| Use case Id(s) | B05 |
| Requirement(s) | B05.FR.01, B05.FR.02, B05.FR.03 |
| System under test | CSMS |
| Description | Set the value of two of the required variables of OCPPCommCtrlr |
| Purpose | To test setting multiple values using SetVariablesRequest for one of the mandatory component/variable combinations that must exist in the DM implementation. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. Test System responds with: SetVariablesResponse | 1. <i>Manually request CSMS to set data for:</i> - OCPPCommCtrlr.OfflineThreshold - AuthCtrlr.AuthorizeRemoteStart+ |

| |
|---|
| Tool validations |
| <p>* Step 1:</p> <p>Message: SetVariablesRequest with (in arbitraty order):</p> <p>setVariableData[1]:</p> <ul style="list-style-type: none"> - variable.name = "OfflineThreshold" - component.name = "OCPPCommCtrlr" - attributeValue = "123" - attributeType is absent or attributeType = <i>Actual</i> <p>setVariableData[2]:</p> <ul style="list-style-type: none"> - variable.name = "AuthorizeRemoteStart" - component.name = "AuthCtrlr" - attributeValue = "false" - attributeType is absent or attributeType = <i>Actual</i> <p>Post scenario validations:</p> <p>Manually validate that CSMS has correctly set the requested variables.</p> |

TC_B_12_CSMS: Get Base Report - ConfigurationInventory

| | |
|-------------------|--|
| Test case name | Get Base Report - ConfigurationInventory |
| Test case Id | TC_B_12_CSMS |
| Use case Id(s) | B07 |
| Requirement(s) | B07.FR.07 |
| System under test | CSMS |
| Description | CSMS requests a ConfigurationInventory base report. |
| Purpose | To test that CSMS supports the ConfigurationInventory base report. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. Test System responds with: GetBaseReportResponse | 1. <i>Manually instruct CSMS to retrieve a ConfigurationInventory report.</i> |

| |
|--|
| Tool validations |
| * Step 1: Message: GetBaseReportRequest with: - requestId has integer value ≥ 0 - reportBase = <i>ConfigurationInventory</i> |
| Post scenario validations: CSMS receives all NotifyReportRequest message for this <i>requestId</i> and is able to show the result of configuration inventory to an operator. |

TC_B_13_CSMS: Get Base Report - FullInventory

| | |
|-------------------|---|
| Test case name | Get Base Report - FullInventory |
| Test case Id | TC_B_13_CSMS |
| Use case Id(s) | B07 |
| Requirement(s) | B07.FR.08 |
| System under test | CSMS |
| Description | CSMS requests a FullInventory base report. |
| Purpose | To test that CSMS supports the FullInventory base report. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. Test System responds with: GetBaseReportResponse | 1. <i>Manually instruct CSMS to retrieve a FullInventory report.</i> |

| |
|---|
| Tool validations |
| * Step 1: GetBaseReportRequest with: - requestId has integer value ≥ 0 - reportBase = <i>FullInventory</i> |
| Post scenario validations: CSMS receives all NotifyReportRequest message for this <i>requestId</i> and is able to show the result of full inventory to an operator. |

TC_B_14_CSMS: Get Base Report - SummaryInventory

| | |
|-------------------|---|
| Test case name | Get Base Report - SummaryInventory |
| Test case Id | TC_B_14_CSMS |
| Use case Id(s) | B07 |
| Requirement(s) | B07.FR.09 |
| System under test | CSMS |
| Description | CSMS requests a SummaryInventory base report. |
| Purpose | To test that CSMS supports the SummaryInventory base report. |
| Prerequisite(s) | CSMS implementation supports the optional SummaryInventory report |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. Test System responds with: GetBaseReportResponse | 1. <i>Manually instruct CSMS to retrieve a SummaryInventory report.</i> |

| |
|---|
| Tool validations |
| * Step 1: GetBaseReportRequest with: - requestId has integer value ≥ 0 - reportBase = <i>SummaryInventory</i> |
| Post scenario validations: CSMS receives all NotifyReportRequest message for this <i>requestId</i> and is able to show the result of summary inventory to an operator. |

TC_B_18_CSMS: Get Custom Report - with componentCriteria and component/variables

| | |
|-------------------|--|
| Test case name | Get Custom Report - with componentCriteria and component/variables |
| Test case Id | TC_B_18_CSMS |
| Use case Id(s) | B08 |
| Requirement(s) | B08.FR.01, B08.FR.03 |
| System under test | CSMS |
| Description | CSMS requests a report of components that match both the component criteria and the given list of components and variables. |
| Purpose | To test that CSMS supports requesting a report for both the component criteria and a given list of components and optionally with variables and that it handles an empty result set. |
| Prerequisite(s) | |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|---|---|
| Charging Station | CSMS |
| 2. Test System responds with: GetReportResponse with status <i>EmptyResultSet</i> | 1. <i>Manually instruct CSMS to get the value of:</i> - EVSE #1::AvailabilityState - from all <i>Problem</i> components |
| 4. Test System responds with: GetReportResponse with status <i>Accepted</i> | 3. <i>Manually instruct CSMS to get the value of:</i> - EVSE #1::AvailabilityState - from all <i>Available</i> components |
| 5. Test System responds with: NotifyReportRequest | 6. CSMS sends NotifyReportResponse |

| |
|--|
| Tool validations |
| * Step 1: Message: GetReportRequest - componentCriteria = <i>Problem</i> - componentVariable[0].component.name = "EVSE" - componentVariable[0].component.evse.id = 1 - componentVariable[0].variable.name = "AvailabilityState" |
| * Step 3: Message: GetReportRequest - componentCriteria is <i>Available</i> - componentVariable[0].component.name = "EVSE" - componentVariable[0].component.evse.id = 1 - componentVariable[0].variable.name = "AvailabilityState" |
| Post scenario validations: N/A |

TC_B_20_CSMS: Reset Charging Station - Without ongoing transaction - OnIdle

| | |
|-------------------|--|
| Test case name | Reset Charging Station - Without ongoing transaction - OnIdle |
| Test case Id | TC_B_20_CSMS |
| Use case Id(s) | B11 |
| Requirement(s) | B11.FR.04 |
| System under test | CSMS |
| Description | This test case covers how the CSMS can request the Charging Station to reset itself by sending a ResetRequest without any ongoing transaction. This could for example be necessary if the Charging Station is not functioning correctly. |
| Purpose | To verify if the CSMS is able to perform the reset mechanism as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action:</u> Request the CSMS to reboot the Charging Station with type_OnIdle | |
| 2. The Test System responds with a ResetResponse with status Accepted | 1. The CSMS sends a ResetRequest |
| 3. The Test System sends a BootNotificationRequest | 4. The CSMS responds with a BootNotificationResponse |
| 5. The Test System notifies the CSMS about the current state of all connectors. Message: StatusNotificationRequest - connectorStatus Available Message: NotifyEventRequest - trigger Delta - actualValue "Available" - component.name "Connector" - variable.name "AvailabilityState" | 6. The CSMS responds accordingly. |

| |
|--|
| Tool validations |
| * Step 1: Message ResetRequest - evseld must be omitted |
| * Step 4: Message BootNotificationResponse - status Accepted |
| Post scenario validations: - N/a |

TC_B_21_CSMS: Reset Charging Station - With Ongoing Transaction - OnIdle

| | |
|-------------------|--|
| Test case name | Reset Charging Station - With Ongoing Transaction - OnIdle |
| Test case Id | TC_B_21_CSMS |
| Use case Id(s) | B12 |
| Requirement(s) | B12.FR.01, B12.FR.03, E07.FR.03 |
| System under test | CSMS |
| Description | <p>This test case covers how the CSMS can remotely request the Charging Station to reset itself by sending a ResetRequest during a transaction. When ResetRequest "OnIdle" is send the charging stations schedules a reboot after all transactions are stopped.</p> <p>This could for example be necessary if the Charging Station is not functioning correctly.</p> |
| Purpose | To verify if the CSMS is able to perform the reset mechanism as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action:</u> Request the CSMS to reboot the Charging Station with status OnIdle | |
| 2. The Test System responds with a ResetResponse with status <i>Scheduled</i> | 1. The CSMS sends a ResetRequest with status <i>OnIdle</i> |
| 3. The Test System sends a TransactionEventRequest . - eventType <i>Updated</i> - triggerReason <i>StopAuthorized</i> - transactionInfo.chargingState <i>EVConnected</i> - idToken.idToken <i><Configured valid_idtoken_idtoken></i> - idToken.type <i><Configured valid_idtoken_type></i> | 4. The CSMS responds with a TransactionEventResponse . |
| 5. The Test System sends a TransactionEventRequest . - eventType <i>Ended</i> - triggerReason <i>EVCommunicationLost</i> - transactionInfo.chargingState <i>Idle</i> - transactionInfo.stoppedReason <i>EVDisconnected</i> | 6. The CSMS responds with a TransactionEventResponse . |
| 7. The Test System sends a BootNotificationRequest with reason <i>ScheduledReset</i> | 8. The CSMS responds with a BootNotificationResponse |
| 9. The Test System notifies the CSMS about the current state of all connectors. Message: StatusNotificationRequest - connectorStatus <i>Available</i> Message: NotifyEventRequest - trigger <i>Delta</i> - actualValue <i>"Available"</i> - component.name <i>"Connector"</i> - variable.name <i>"AvailabilityState"</i> | 10. The CSMS responds accordingly. |

| Tool validations |
|---|
| <p>* Step 1:</p> <p>Message ResetRequest</p> <ul style="list-style-type: none">- type <i>OnIdle</i>- evseld must be omitted <p>* Step 8:</p> <p>Message BootNotificationResponse</p> <ul style="list-style-type: none">- status <i>Accepted</i> |
| <p>Post scenario validations:</p> <ul style="list-style-type: none">- N/a |

TC_B_22_CSMS: Reset Charging Station - With Ongoing Transaction - Immediate

| | |
|-------------------|--|
| Test case name | Reset Charging Station - With Ongoing Transaction - Immediate |
| Test case Id | TC_B_22_CSMS |
| Use case Id(s) | B12 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | <p>This test case covers how the CSMS can remotely request the Charging Station to reset itself by sending a ResetRequest during a transaction. When ResetRequest "Immediate" is send the charging stations will try to stop all transactions before rebooting.</p> <p>This could for example be necessary if the Charging Station is not functioning correctly.</p> |
| Purpose | To verify if the CSMS is able to perform the reset mechanism as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> |

| Main (Test scenario) | |
|--|---|
| Charging Station | CSMS |
| <u>Manual Action:</u> Request the CSMS to reboot the Charging Station with status Immediate | |
| 2. The Test System responds with a ResetResponse with status <i>Accepted</i> | 1. The CSMS sends a ResetRequest with status <i>Immediate</i> |
| 3. The Test System sends a TransactionEventRequest . - eventType <i>Ended</i> - triggerReason <i>ResetCommand</i> - transactionInfo.chargingState <i>EVConnected</i> - transactionInfo.stoppedReason <i>ImmediateReset</i> - idToken is omitted | 4. The CSMS responds with a TransactionEventResponse . |
| 5. The Test System sends a BootNotificationRequest with reason <i>RemoteReset</i> | 6. The CSMS responds with a BootNotificationResponse |

| Main (Test scenario) | |
|--|--|
| <p>7. The Test System notifies the CSMS about the current state of all connectors.</p> <p>For <Configured connectorId>:</p> <p>Message: StatusNotificationRequest</p> <ul style="list-style-type: none">- connectorStatus <i>Occupied</i> <p>Message: NotifyEventRequest</p> <ul style="list-style-type: none">- trigger <i>Delta</i>- actualValue <i>"Occupied"</i>- component.name <i>"Connector"</i>- variable.name <i>"AvailabilityState"</i> <p>For <Other connector(s)>:</p> <p>Message: StatusNotificationRequest</p> <ul style="list-style-type: none">- connectorStatus <i>Available</i> <p>Message: NotifyEventRequest</p> <ul style="list-style-type: none">- trigger <i>Delta</i>- actualValue <i>"Available"</i>- component.name <i>"Connector"</i>- variable.name <i>"AvailabilityState"</i> | <p>8. The CSMS responds accordingly.</p> |
| Tool validations | |
| <p>* Step 1:</p> <p>Message ResetRequest</p> <ul style="list-style-type: none">- type <i>Immediate</i>- evseld must be omitted <p>* Step 6:</p> <p>Message BootNotificationResponse</p> <ul style="list-style-type: none">- status <i>Accepted</i> | |
| <p>Post scenario validations:</p> <ul style="list-style-type: none">- N/a | |

TC_B_25_CSMS: Reset EVSE - Without ongoing transaction

| | |
|-------------------|---|
| Test case name | Reset EVSE - Without ongoing transaction |
| Test case Id | TC_B_25_CSMS |
| Use case Id(s) | B11 |
| Requirement(s) | B11.FR.04 |
| System under test | CSMS |
| Description | This test case covers how the CSMS can request the Charging Station to reset an EVSE by sending a ResetRequest without any ongoing transaction. This could for example be necessary if the Charging Station is not functioning correctly. |
| Purpose | To verify if the CSMS is able to perform the reset mechanism as described at the OCPP specification. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action:</u> Request the CSMS to reboot an EVSE with status OnIdle | |
| 2. The Test System responds with a ResetResponse with status Accepted | 1. The CSMS sends a ResetRequest with status OnIdle and evsID <Configured evseld> |

| |
|---|
| Tool validations |
| * Step 1: Message ResetRequest - type OnIdle - evseld <Configured evseld> |
| Post scenario validations: - N/a |

TC_B_26_CSMS: Reset EVSE - With Ongoing Transaction - OnIdle

| | |
|-------------------|---|
| Test case name | Reset EVSE - With Ongoing Transaction - OnIdle |
| Test case Id | TC_B_26_CSMS |
| Use case Id(s) | B12 |
| Requirement(s) | B12.FR.07 |
| System under test | CSMS |
| Description | <p>This test case covers how the CSMS can remotely request the Charging Station to reset an EVSE by sending a ResetRequest during a transaction. When ResetRequest "OnIdle" is send the charging stations schedules a reboot after all transactions are stopped.</p> <p>This could for example be necessary if the Charging Station is not functioning correctly.</p> |
| Purpose | To verify if the CSMS is able to perform the reset mechanism as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action:</u> Request the CSMS to reboot the charging EVSE with status OnIdle | |
| 2. The Test System responds with a ResetResponse with status <i>Scheduled</i> | 1. The CSMS sends a ResetRequest with status <i>OnIdle</i> and evseld <Configured evseld> |
| 3. The Test System sends a TransactionEventRequest . - eventType <i>Updated</i> - triggerReason <i>StopAuthorized</i> - transactionInfo.chargingState <i>EVConnected</i> - idToken.idToken <Configured valid_idtoken_idtoken> - idToken.type <Configured valid_idtoken_type> | 4. The CSMS responds with a TransactionEventResponse . |
| 5. The Test System sends a TransactionEventRequest . - eventType <i>Ended</i> - triggerReason <i>EVCommunicationLost</i> - transactionInfo.chargingState <i>Idle</i> - transactionInfo.stoppedReason <i>EVDisconnected</i> | 6. The CSMS responds with a TransactionEventResponse . |

| |
|--|
| Tool validations |
| * Step 1: Message ResetRequest - type <i>OnIdle</i> - evseld <Configured evseld> |
| Post scenario validations: - N/a |

TC_B_27_CSMS: Reset EVSE - With Ongoing Transaction - Immediate

| | |
|-------------------|---|
| Test case name | Reset EVSE - With Ongoing Transaction - Immediate |
| Test case Id | TC_B_27_CSMS |
| Use case Id(s) | B12 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | <p>This test case covers how the CSMS can remotely request the Charging Station to reset an EVSE by sending a ResetRequest during a transaction. When ResetRequest "Immediate" is send the charging stations will try to stop all transactions before rebooting.</p> <p>This could for example be necessary if the Charging Station is not functioning correctly.</p> |
| Purpose | To verify if the CSMS is able to perform the reset mechanism as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> |

| Main (Test scenario) | |
|--|---|
| Charging Station | CSMS |
| <u>Manual Action:</u> Request the CSMS to reboot the charging EVSE with status Immediate | |
| 2. The Test System responds with a ResetResponse with status Accepted | 1. The CSMS sends a ResetRequest with status Immediate and evseld <Configured evseld> |
| 3. The Test System sends a TransactionEventRequest . - eventType Ended - triggerReason ResetCommand - transactionInfo.chargingState EVConnected - transactionInfo.stoppedReason ImmediateReset | 4. The CSMS responds with a TransactionEventResponse . |

| |
|--|
| Tool validations |
| * Step 1: Message ResetRequest - type <i>Immediate</i> - evseld <i><Configured evseld></i> |
| Post scenario validations: N/a |

TC_B_42_CSMS: Set new NetworkConnectionProfile - Accepted

| | |
|-------------------|--|
| Test case name | Set new NetworkConnectionProfile - Accepted |
| Test case Id | TC_B_42_CSMS |
| Use case Id(s) | B09 |
| Requirement(s) | B09.FR.01 |
| System under test | CSMS |
| Description | The CSMS updates the connection details on the Charging Station. For instance in preparation of a migration to a new CSMS. |
| Purpose | To verify if the CSMS is able to set a new network connection profile at one of the by the Charging Station defined configuration slots. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a SetNetworkProfileResponse With status <i>Accepted</i> | 1. The CSMS sends a SetNetworkProfileRequest |

| |
|--|
| Tool validations |
| <p>* Step 1:</p> <p>Message SetNetworkProfileRequest</p> <ul style="list-style-type: none"> - configurationSlot is <Configured configurationSlot> - connectionData.messageTimeout <Configured messageTimeout> - connectionData.ocppCsmsUrl <Configured ocppCsmsUrl> - connectionData.ocppInterface <Configured ocppInterface> - connectionData.ocppTransport JSON - connectionData.ocppVersion OCPP20 - connectionData.securityProfile <Configured securityProfile> |
| Post scenario validations: |
| - N/a |

TC_B_44_CSMS: Set new NetworkConnectionProfile - Failed

| | |
|--------------------------|--|
| Test case name | Set new NetworkConnectionProfile - Failed |
| Test case Id | TC_B_44_CSMS |
| Use case Id(s) | B09 |
| Requirement(s) | B09.FR.03 |
| System under test | CSMS |
| Description | The CSMS updates the connection details on the Charging Station. For instance in preparation of a migration to a new CSMS. |
| Purpose | To verify if the CSMS is able to handle a Charging Station responding with status Failed, when setting a new network connection profile at one of the by the Charging Station defined configuration slots. |
| Prerequisite(s) | N/a |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a SetNetworkProfileResponse With status Failed | 1. The CSMS sends a SetNetworkProfileRequest |

| |
|--|
| Tool validations |
| N/a |
| Post scenario validations: - N/a |

TC_B_58_CSMS: WebSocket Subprotocol negotiation

| | |
|-------------------|--|
| Test case name | WebSocket Subprotocol validation |
| Test case Id | TC_B_58_CSMS |
| Use case Id(s) | Part 4 - JSON over WebSockets implementation guide |
| Requirement(s) | Section 3.1.2. OCPP version |
| System under test | CSMS |
| Description | OCPP-J imposes extra constraints on the WebSocket subprotocol, detailed in the following section 3.1.2. |
| Purpose | To verify whether the CSMS is able to select a supported OCPP version, when also a different unsupported version is supported by the Charging Station and relays this selection via the Sec-Websocket-Protocol header. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Test System disconnects the WebSocket connection and reconnects by sending a HTTP upgrade request with the header; Sec-WebSocket-Protocol: ocpp0.1 | 2. The CSMS rejects the connection attempt and does NOT upgrade the connection to a WebSocket connection. |
| 3. The Test System disconnects the WebSocket connection and reconnects by sending a HTTP upgrade request with the header; Sec-WebSocket-Protocol: ocpp0.1,ocpp<Selected OCPP version> | 4. The CSMS accepts the connection attempt and upgrades the connection to a WebSocket connection. |

| |
|--|
| Tool validations |
| * Step 4: The authorization header of the HTTP upgrade response must contain the header Sec-Websocket-Protocol, and it must comply to the following: - The header is formatted as follows; Sec-WebSocket-Protocol: ocpp<Selected OCPP version> |
| Post scenario validations: N/a |

C Authorization

TC_C_02_CSMS: Local start transaction - Authorization Invalid/Unknown

| | |
|-------------------|---|
| Test case name | Local start transaction - Authorization Invalid/Unknown |
| Test case Id | TC_C_02_CSMS |
| Use case Id(s) | C01, C04, C06 |
| Requirement(s) | C01.FR.07 OR C04.FR.01 OR C06.FR.04 |
| System under test | CSMS |
| Description | When a Charging Station needs to charge an EV, it needs to authorize the EV Driver first at the CSMS before the charging can be started or stopped. |
| Purpose | To verify whether the CSMS is able to report that an idToken is NOT valid. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Test System sends an AuthorizeRequest with idToken.idToken <Configured invalid_idtoken_idtoken> idToken.type <Configured invalid_idtoken_type> | 2. The CSMS responds with an AuthorizeResponse |

| |
|--|
| Tool validations |
| * Step 2: Message: AuthorizeResponse - idTokenInfo.status <i>Invalid</i> or <i>Unknown</i> |
| Post scenario validations: - N/a |

TC_C_06_CSMS: Local start transaction - Authorization Blocked

| | |
|-------------------|---|
| Test case name | Local start transaction - Authorization Blocked |
| Test case Id | TC_C_06_CSMS |
| Use case Id(s) | C01 |
| Requirement(s) | C01.FR.07 |
| System under test | CSMS |
| Description | When a Charging Station needs to charge an EV, it needs to authorize the EV Driver first at the CSMS before the charging can be started or stopped. |
| Purpose | To verify whether the CSMS is able to report that an idToken is Blocked. |
| Prerequisite(s) | N/a |

| |
|--|
| Before (Preparations) |
| Configuration State: The IdToken configured as Blocked at the Test System, must be set as Blocked at the CSMS. |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Test System sends an AuthorizeRequest with idToken.idToken <Configured blocked_idtoken_idtoken> idToken.type <Configured blocked_idtoken_type> | 2. The CSMS responds with an AuthorizeResponse |

| |
|--|
| Tool validations |
| * Step 2: Message: AuthorizeResponse - idTokenInfo.status <i>Blocked</i> or <i>Invalid</i> |
| Post scenario validations: |

TC_C_07_CSMS: Local start transaction - Authorization Expired

| | |
|-------------------|---|
| Test case name | Local start transaction - Authorization Expired |
| Test case Id | TC_C_07_CSMS |
| Use case Id(s) | C01 |
| Requirement(s) | C01.FR.07 |
| System under test | CSMS |
| Description | When a Charging Station needs to charge an EV, it needs to authorize the EV Driver first at the CSMS before the charging can be started or stopped. |
| Purpose | To verify whether the CSMS is able to report that an idToken is Expired. |
| Prerequisite(s) | N/a |

| |
|--|
| Before (Preparations) |
| Configuration State: The IdToken configured as Expired at the Test System, must be set as Expired at the CSMS. |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Test System sends an AuthorizeRequest with idToken.idToken <Configured expired_idtoken_idtoken> idToken.type <Configured expired_idtoken_type> | 2. The CSMS responds with an AuthorizeResponse |

| |
|--|
| Tool validations |
| * Step 2: Message: AuthorizeResponse - idTokenInfo.status Expired or Invalid |
| Post scenario validations: |

TC_C_08_CSMS: Authorization through authorization cache - Accepted

| | |
|-------------------|---|
| Test case name | Authorization through authorization cache - Accepted |
| Test case Id | TC_C_08_CSMS |
| Use case Id(s) | C12 |
| Requirement(s) | C12_FR_03 |
| System under test | CSMS |
| Description | This test case describes how the EV Driver is authorized to start a transaction while the Charging Station uses Cached IdToken. This enables the EV Driver to Online start a transaction by using the Authorization Cache in which the Charging Station can respond faster, as no AuthorizeRequest is being sent. |
| Purpose | To verify if the CSMS is able to respond correctly when an idToken which has status "Accepted" in the charging stations cache is presented according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | N/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: State is <i>EVConnectedPreSession</i> |

| Main (Test scenario) | |
|--|--|
| Charging Station | CSMS |
| <div>1. The Test System sends a TransactionEventRequest with</div> <div><div>- triggerReason <i>Authorized</i></div><div>- idToken <i><Valid id token configured in Authorization Cache></i></div><div>- eventType <i>Updated</i></div></div> <div><div>Note(s):</div><div>- TxStartPoint <i>contains ParkingBayOccupancy</i></div></div> | <div>2. The CSMS responds with a TransactionEventResponse</div> |

| |
|---|
| Tool validations |
| * Step 2: Message TransactionEventResponse - idTokenInfo.status <i>Accepted</i> |
| Post scenario validations: - N/a |

TC_C_20_CSMS: Authorization through authorization cache - Invalid

| | |
|-------------------|---|
| Test case name | Authorization through authorization cache - Invalid |
| Test case Id | TC_C_20_CSMS |
| Use case Id(s) | C12 |
| Requirement(s) | C12_FR_03 |
| System under test | CSMS |
| Description | This test case describes how the EV Driver is authorized to start a transaction while the Charging Station uses Cached IdToken. This enables the EV Driver to Online start a transaction by using the Authorization Cache in which the Charging Station can respond faster, as no AuthorizeRequest is being sent. |
| Purpose | To verify if the CSMS is able to respond correctly when an idToken, which has status "Invalid" in the charging stations cache but not in the CSMS, is presented according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | N/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: State is <i>EVConnectedPreSession</i> |

| Main (Test scenario) | |
|---|--|
| Charging Station | CSMS |
| <p>1. The Test System sends a TransactionEventRequest with</p> <ul style="list-style-type: none">- triggerReason <i>Authorized</i>- idToken.idToken <i><Configured invalid_idtoken_idtoken></i>- idToken.type <i><Configured invalid_idtoken_type></i> - eventType <i>Updated</i> <p><u>Note(s):</u></p> <ul style="list-style-type: none">- TxStartPoint contains <i>ParkingBayOccupancy</i> | <p>2. The CSMS responds with a TransactionEventResponse</p> |

| |
|--|
| Tool validations |
| * Step 2: Message TransactionEventResponse - idTokenInfo.status <i>Invalid</i> or <i>Unknown</i> |
| Post scenario validations: - N/a |

TC_C_37_CSMS: Clear Authorization Data in Authorization Cache - Accepted

| | |
|-------------------|---|
| Test case name | Clear Authorization Data in Authorization Cache - Accepted |
| Test case Id | TC_C_37_CSMS |
| Use case Id(s) | C11 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | This test case covers how the Charging Station autonomously stores a record of previously presented identifiers that have been successfully authorized by the CSMS in the Authorization Cache. (Successfully meaning: a response received on a message containing an IdToken) |
| Purpose | To verify if the CSMS is able to request the Charging Station to clear all identifiers from the Authorization Cache according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | - N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a ClearCacheResponse with status <i>Accepted</i> | 1. The CSMS sends a ClearCacheRequest |

| |
|-------------------------------------|
| Tool validations |
| - N/a |
| Post scenario validations: - N/a |

TC_C_38_CSMS: Clear Authorization Data in Authorization Cache - Rejected

| | |
|-------------------|---|
| Test case name | Clear Authorization Data in Authorization Cache - Rejected |
| Test case Id | TC_C_38_CSMS |
| Use case Id(s) | C11 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | This test case covers how the Charging Station autonomously stores a record of previously presented identifiers that have been successfully authorized by the CSMS in the Authorization Cache. (Successfully meaning: a response received on a message containing an IdToken) |
| Purpose | To verify if the CSMS is able to request the Charging Station to clear all identifiers from the Authorization Cache according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | - N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a ClearCacheResponse with status <i>Rejected</i> | 1. The CSMS sends a ClearCacheRequest |

| |
|-------------------------------------|
| Tool validations |
| - N/a |
| Post scenario validations: - N/a |

TC_C_39_CSMS: Authorization by GroupId - Success

| | |
|-------------------|---|
| Test case name | Authorization by GroupId - Success |
| Test case Id | TC_C_39_CSMS |
| Use case Id(s) | C09 |
| Requirement(s) | C09_FR_02, C09_FR_03 |
| System under test | CSMS |
| Description | This test case covers how a Charging Station can authorize an action for an EV Driver based on GroupId information. This could for example be used if 2 people regularly use the same EV: they can use their own IdToken (e.g. RFID card), and can deauthorize transactions that were started with the other idToken (with the same GroupId). |
| Purpose | To verify if the CSMS is able to correctly handle the Authorization of idTokens with the same GroupId according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | - N/a |

Before (Preparations)

Configuration State:

N/a

Memory State:

Two valid idTokens with the same GroupId are configured

Reusable State(s):

state is [EVConnectedPreSession](#)

Main (Test scenario)

| Charging Station | CSMS |
|---|---|
| 1. The Test System sends an AuthorizeRequest with idToken.idToken <Configured valid_idtoken2_idtoken> idToken.type <Configured valid_idtoken2_type> | 2. The CSMS responds with an AuthorizeResponse |
| 3. The Test System sends a TransactionEventRequest with - triggerReason <i>Authorized</i> - idToken.idToken <Configured valid_idtoken_idtoken> - idToken.type <Configured valid_idtoken_type> if transaction was already started - eventType <i>Updated</i> else - eventType <i>Started</i> | 4. The CSMS responds with a TransactionEventResponse |
| 5. Execute Reusable State EnergyTransferStarted | |
| 6. The Test System sends an AuthorizeRequest with idToken.idToken <Configured valid_idtoken2_idtoken> idToken.type <Configured valid_idtoken2_type> | 7. The CSMS responds with an AuthorizeResponse |
| 8. The Test System sends a TransactionEventRequest with - triggerReason <i>StopAuthorized</i> - idToken.idToken <Configured valid_idtoken2_idtoken> - idToken.type <Configured valid_idtoken2_type> - eventType <i>Updated</i> | 9. The CSMS responds with a TransactionEventResponse |
| 10. Execute Reusable State EVConnectedPostSession | |
| 11. Execute Reusable State EVDisconnected | |

| Tool validations |
|---|
| <p>* Step 2:</p> <p>Message AuthorizeResponse</p> <ul style="list-style-type: none">- idTokenInfo.status <i>Accepted</i>- idTokenInfo.groupIdToken.idToken <i><Configured groupIdToken></i> <p>* Step 4:</p> <p>Message TransactionEventResponse</p> <ul style="list-style-type: none">- idTokenInfo.status <i>Accepted</i>- idTokenInfo.groupIdToken.idToken <i><Configured groupIdToken></i> <p>* Step 7:</p> <p>Message AuthorizeResponse</p> <ul style="list-style-type: none">- idTokenInfo.status <i>Accepted</i>- idTokenInfo.groupIdToken.idToken <i><Configured groupIdToken></i> <p>* Step 9:</p> <p>Message TransactionEventResponse</p> <ul style="list-style-type: none">- idTokenInfo.status <i>Accepted</i>- idTokenInfo.groupIdToken.idToken <i><Configured groupIdToken></i> |
| <p>Post scenario validations:</p> <ul style="list-style-type: none">- N/a |

TC_C_40_CSMS: Authorization by GroupId - Success with Local Authorization List

| | |
|-------------------|---|
| Test case name | Authorization by GroupId - Success with Local Authorization List |
| Test case Id | TC_C_40_CSMS |
| Use case Id(s) | C09 |
| Requirement(s) | C09_FR_02, C09_FR_03 |
| System under test | CSMS |
| Description | This test case covers how a Charging Station can authorize an action for an EV Driver based on GroupId information. This could for example be used if 2 people regularly use the same EV: they can use their own IdToken (e.g. RFID card), and can deauthorize transactions that were started with the other idToken (with the same GroupId). |
| Purpose | To verify if the CSMS is able to correctly handle the Authorization of idTokens with the same GroupId which are located in the Local Authorization List according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | - N/a |

Before (Preparations)

Configuration State:

N/a

Memory State:

Two valid idTokens with same GroupId are configured

Reusable State(s):

state is *EVConnectedPreSession*

Main (Test scenario)

| Charging Station | CSMS |
|---|--|
| 1. The Test System sends a TransactionEventRequest with - triggerReason <i>Authorized</i> - idToken.idToken <i><Configured valid_idtoken_idtoken></i> (with a configured GroupId) which is configured in the local Authorization List - idToken.type <i><Configured valid_idtoken_type></i> (with a configured GroupId) which is configured in the local Authorization List If transaction was already started - eventType <i>Updated</i> else - eventType <i>Started</i> | 2. The CSMS responds with a TransactionEventResponse |
| 3. Execute Reusable State <i>EnergyTransferStarted</i> | |
| 5. The Test System sends a TransactionEventRequest with - triggerReason <i>StopAuthorized</i> - idToken.idToken <i><Configured valid_idtoken2_idtoken></i> (with same configured GroupId) which is configured in the local Authorization List - idToken.type <i><Configured valid_idtoken2_type></i> - eventType <i>Updated</i> | 6. The CSMS responds with a TransactionEventResponse |
| 7. Execute Reusable State <i>EVConnectedPostSession</i> | |
| 8. Execute Reusable State <i>EVDisconnected</i> | |

| Tool validations |
|---|
| <p>* Step 2:</p> <p>Message TransactionEventResponse</p> <ul style="list-style-type: none">- idTokenInfo.status <i>Accepted</i>- idTokenInfo.groupIdToken.idToken <i><Configured groupIdToken></i> <p>* Step 6:</p> <p>Message TransactionEventResponse</p> <ul style="list-style-type: none">- idTokenInfo.status <i>Accepted</i>- idTokenInfo.groupIdToken.idToken <i><Configured groupIdToken></i> |
| <p>Post scenario validations:</p> <ul style="list-style-type: none">- N/a |

TC_C_43_CSMS: Authorization by GroupId - Invalid status with Local Authorization List

| | |
|-------------------|---|
| Test case name | Authorization by GroupId - Invalid status with Local Authorization List |
| Test case Id | TC_C_43_CSMS |
| Use case Id(s) | C09 |
| Requirement(s) | C09_FR_02, C09_FR_03 |
| System under test | CSMS |
| Description | This test case covers how a Charging Station can authorize an action for an EV Driver based on GroupId information. This could for example be used if 2 people regularly use the same EV: they can use their own IdToken (e.g. RFID card), and can deauthorize transactions that were started with the other idToken (with the same GroupId). |
| Purpose | To verify if the CSMS is able to correctly handle the Authorization of idTokens with the same GroupId which are located in the Local Authorization List according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | - N/a |

Before (Preparations)

Configuration State:

N/a

Memory State:

Two known valid idTokens with same GroupId are configured.

Reusable State(s):

state is [EVConnectedPreSession](#)

Main (Test scenario)

| Charging Station | CSMS |
|---|--|
| 1. The Test System sends a TransactionEventRequest with - triggerReason <i>Authorized</i> - idToken.idToken <i><Configured valid_idtoken_idtoken></i> - idToken.type <i><Configured valid_idtoken_type></i> if transaction was already started - eventType <i>Updated</i> else - eventType <i>Started</i> | 2. The CSMS responds with a TransactionEventResponse |
| 3. Execute Reusable State EnergyTransferStarted | |
| 4. The Test System sends an AuthorizeRequest with - idToken.idToken <i><Configured valid_idtoken2_idtoken></i> - idToken.type <i><Configured valid_idtoken2_type></i> | 5. The CSMS responds with an AuthorizeResponse |
| 6. The Test System sends a TransactionEventRequest with - triggerReason <i>StopAuthorized</i> - idToken.idToken <i><Configured valid_idtoken2_idtoken></i> - idToken.type <i><Configured valid_idtoken2_type></i> - eventType <i>Updated</i> | 7. The CSMS responds with a TransactionEventResponse |
| 8. Execute Reusable State EVConnectedPostSession | |
| 9. Execute Reusable State EVDisconnected | |

| Tool validations |
|---|
| <p>* Step 1:</p> <p>Message TransactionEventResponse</p> <ul style="list-style-type: none">- idTokenInfo.status <i>Accepted</i>- idTokenInfo.groupIdToken.idToken <i><Configured groupIdToken></i> <p>* Step 5:</p> <p>Message AuthorizeResponse</p> <ul style="list-style-type: none">- idTokenInfo.status <i>Accepted</i>- idTokenInfo.groupIdToken.idToken <i><Configured groupIdToken></i> <p>* Step 7:</p> <p>Message TransactionEventResponse</p> <ul style="list-style-type: none">- idTokenInfo.status <i>Accepted</i>- idTokenInfo.groupIdToken.idToken <i><Configured groupIdToken></i> |
| <p>Post scenario validations:</p> <ul style="list-style-type: none">- N/a |

TC_C_47_CSMS: Stop Transaction with a Master Pass - With UI - All transactions

| | |
|-------------------|--|
| Test case name | Stop Transaction with a Master Pass - With UI - All transactions |
| Test case Id | TC_C_47_CSMS |
| Use case Id(s) | C16 |
| Requirement(s) | C16_FR_01 |
| System under test | CSMS |
| Description | This test case covers how somebody with a Master Pass (User) can stop (selected) ongoing transactions, so the cable becomes unlocked. This Master Pass can be configured in: MasterPassGroupId. This could for example be usefull for Law Enforcement officials. |
| Purpose | To verify if the CSMS is able to correctly respond on a request to stop all transactions when an idToken which has the MasterPass as GroupId is used and the user has selected to stop all transactions in the User Interface according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | - N/a |

Before (Preparations)

Configuration State:

N/a

Memory State:

An idToken with the MastersPass as GroupId is configured

Reusable State(s):

State is *EnergyTransferStarted* for EVSE 1 with idToken valid idTokenState is *EnergyTransferStarted* for EVSE 2 with idToken valid idToken2

Main (Test scenario)

| Charging Station | CSMS |
|--|---|
| 1. The Test System sends an AuthorizeRequest with idToken.idToken <Configured masterpass_idtoken_idtoken> idToken.type <Configured masterpass_idtoken_type> | 2. The CSMS responds with an AuthorizeResponse |
| 3. The Test System sends a TransactionEventRequest with - transactionInfo.stoppedReason MasterPass - idToken.idToken <Configured masterpass_idtoken_idtoken> - idToken.type <Configured masterpass_idtoken_type> - eventType Ended for both EVSE | 4. The CSMS responds with a TransactionEventResponse for both EVSE |

Tool validations

* Step 2:

Message **AuthorizeResponse**- **idTokenInfo.status** Accepted- **idTokenInfo.groupIdToken.idToken** <Configured masterPassGroupId>

* Step 4:

Message **TransactionEventResponse**- **idTokenInfo.status** Accepted- **idTokenInfo.groupIdToken.idToken** <Configured masterPassGroupId>

Post scenario validations:

- N/a

TC_C_48_CSMS: Stop Transaction with a Master Pass - With UI - With UI - Specific transactions

| | |
|--------------------------|--|
| Test case name | Stop Transaction with a Master Pass - With UI - With UI - Specific transactions |
| Test case Id | TC_C_48_CSMS |
| Use case Id(s) | C16 |
| Requirement(s) | C16_FR_01 |
| System under test | CSMS |
| Description | This test case covers how somebody with a Master Pass (User) can stop (selected) ongoing transactions, so the cable becomes unlocked. This Master Pass can be configured in: MasterPassGroupId. This could for example be usefull for Law Enforcement officials. |
| Purpose | To verify if the CSMS is able to correctly respond on a request to stop a transaction when an idToken which has the MasterPass as GroupId is used and the user has selected to stop one transaction in the User Interface according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | - N/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: An idToken with the MastersPass as GroupId is configured |
| Reusable State(s): State is <i>EnergyTransferStarted</i> for all EVSE |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Test System sends an AuthorizeRequest with idToken.idToken <Configured valid_idtoken_idtoken> idToken.type <Configured valid_idtoken_type> | 2. The CSMS responds with an AuthorizeResponse |
| 3. The Test System sends a TransactionEventRequest with - transactionInfo.stoppedReason MasterPass - idToken.idToken <Configured masterpass_idtoken_idtoken> - idToken.type <Configured masterpass_idtoken_type> - eventType Ended | 4. The CSMS responds with a TransactionEventResponse |

| |
|--|
| Tool validations |
| <p>* Step 2: Message AuthorizeResponse - idTokenInfo.status Accepted - idTokenInfo.groupIdToken.idToken <Configured masterPassGroupId></p> <p>* Step 4: Message TransactionEventResponse - idTokenInfo.status Accepted - idTokenInfo.groupIdToken.idToken <Configured masterPassGroupId></p> |
| Post scenario validations: - N/a |

TC_C_49_CSMS: Stop Transaction with a Master Pass - Without UI

| | |
|-------------------|---|
| Test case name | Stop Transaction with a Master Pass - Without UI |
| Test case Id | TC_C_49_CSMS |
| Use case Id(s) | C16 |
| Requirement(s) | C16_FR_02 |
| System under test | CSMS |
| Description | This test case covers how somebody with a Master Pass (User) can stop (selected) ongoing transactions, so the cable becomes unlocked. This Master Pass can be configured in: MasterPassGroupId. This could for example be usefull for Law Enforcement officials. |
| Purpose | To verify if the CSMS is able to correctly respond on a request to stop all transactions when an idToken which has the MasterPass as GroupId is used and the Charging Station does not have a User Interface according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | - N/a |

Before (Preparations)

Configuration State:

N/a

Memory State:

An idToken with the MastersPass as GroupId is configured

Reusable State(s):

State is *EnergyTransferStarted* for EVSE 1 with idToken valid idTokenState is *EnergyTransferStarted* for EVSE 2 with idToken valid idToken2

Main (Test scenario)

| Charging Station | CSMS |
|--|---|
| 1. The Test System sends an AuthorizeRequest with idToken.idToken <Configured masterpass_idtoken_idtoken> idToken.type <Configured masterpass_idtoken_type> | 2. The CSMS responds with an AuthorizeResponse |
| 3. The Test System sends a TransactionEventRequest with - transactionInfo.stoppedReason MasterPass - idToken.idToken <Configured valid_idtoken_idtoken> - idToken.type <Configured valid_idtoken_type> - eventType Ended for both EVSE | 4. The CSMS responds with a TransactionEventResponse for both EVSE |

Tool validations

* Step 2:

Message **AuthorizeResponse**- **idTokenInfo.status** Accepted- **idTokenInfo.groupIdToken.idToken** <Configured masterPassGroupId>

* Step 4:

Message **TransactionEventResponse**- **idTokenInfo.status** Accepted- **idTokenInfo.groupIdToken.idToken** <Configured masterPassGroupId>

Post scenario validations:

- N/a

TC_C_50_CSMS: Authorization using Contract Certificates 15118 - Online - Local contract certificate validation - Accepted

| | |
|-------------------|--|
| Test case name | Authorization using Contract Certificates 15118 - Online - Local contract certificate validation - Accepted |
| Test case Id | TC_C_50_CSMS |
| Use case Id(s) | C07 |
| Requirement(s) | C07.FR.04 |
| System under test | CSMS |
| Description | The Charging Station is able to authorize with contract certificates when it supports ISO 15118. |
| Purpose | To verify if the CSMS is able to validate the certificate hash data and the provided eMAID. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The configured eMAID is known by the CSMS as valid. - The configured contract certificate is valid. - The CN of the configured contract certificate equals the configured eMAID. - iso15118CertificateHashData has a responder URL that points to an OCSP service for Test System. - CSMS does not have a cached OCSP response for the contract certificate. |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is EVConnectedPreSession |

| Main (Test scenario) | |
|---|--|
| Charging Station | CSMS |
| 1. The Test System sends an AuthorizeRequest With idToken.idToken <Configured eMAID> idToken.type eMAID iso15118CertificateHashData contains <hashes from configured (V2G) certificate chain | 2. The CSMS sends an OCSP request to responder URL of iso15118CertificateHashData to check validity |
| 3. The Test System OCSP service reponds that certificate is valid. | 4. The CSMS responds with a AuthorizeResponse |
| 5. The Test System sends a TransactionEventRequest With triggerReason Authorized | 6. The CSMS responds with a TransactionEventResponse |
| 5. Execute Reusable State <i>EnergyTransferStarted</i> | |

| |
|--|
| Tool validations |
| * Step 2: CSMS sends an OCSP request for iso15118CertificateHashData * Step 3: Test System checks that received request for iso15118CertificateHashData is valid * Step 4: Message: AuthorizeResponse - idTokenInfo.status Accepted - certificateStatus Accepted * Step 4: Message: TransactionEventResponse - idTokenInfo.status Accepted |
| Post scenario validations: N/a |

TC_C_51_CSMS: Authorization using Contract Certificates 15118 - Online - Local contract certificate validation - Rejected

| | |
|-------------------|---|
| Test case name | Authorization using Contract Certificates 15118 - Online - Local contract certificate validation - Rejected |
| Test case Id | TC_C_51_CSMS |
| Use case Id(s) | C07 |
| Requirement(s) | C07.FR.16 |
| System under test | CSMS |
| Description | The Charging Station is able to authorize with contract certificates when it supports ISO 15118. |
| Purpose | To verify if the CSMS is able to validate the certificate hash data and the provided eMAID. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The configured eMAID is known by the CSMS as valid. - The contract certificate is revoked. - iso15118CertificateHashData has a responder URL that points to an OCSP service for Test System. - CSMS does not have a cached OCSP response for the contract certificate. |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EVConnectedPreSession</i> |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Test System sends an AuthorizeRequest With idToken.idToken <Configured valid_idtoken_idtoken> idToken.type <Configured valid_idtoken_type> iso15118CertificateHashData contains <hashes from configured (V2G) certificate chain> | 2. The CSMS sends an OCSP request to responder URL of iso15118CertificateHashData to check validity |
| 3. The Test System OCSP service responds that certificate is valid. | 4. The CSMS responds with a AuthorizeResponse |

| |
|--|
| Tool validations |
| * Step 2: CSMS sends an OCSP request for iso15118CertificateHashData * Step 3: Test System checks that received request for iso15118CertificateHashData is valid * Step 4: Message: AuthorizeResponse - idTokenInfo.status <i>Invalid</i> - certificateStatus <i>CertificateRevoked</i> |
| Post scenario validations: EV is not authorized and shall not charge: Charging Station does not send TransactionEventRequest with: - triggerReason = <i>Authorized</i> or chargingState = <i>Charging</i> |

TC_C_52_CSMS: Authorization using Contract Certificates 15118 - Online - Central contract certificate validation - Accepted

| | |
|-------------------|--|
| Test case name | Authorization using Contract Certificates 15118 - Online - Central contract certificate validation - Accepted |
| Test case Id | TC_C_52_CSMS |
| Use case Id(s) | C07 |
| Requirement(s) | C07.FR.04,C07.FR.05 |
| System under test | CSMS |
| Description | The Charging Station is able to authorize with contract certificates when it supports ISO 15118. |
| Purpose | To verify if the CSMS is able to validate the provided certificate and eMAID. The field iso15118CertificateHashData is not provided to force CSMS to calculate certificate hash data for the OCSP request. |
| Prerequisite(s) | <ul style="list-style-type: none"> - The configured eMAID is known by the CSMS as valid. - The contract certificate is signed by the configured V2GRoot or MORoot certificate at the CSMS. - Contract certificate has a responder URL that points to an OCSP service for Test System. - CSMS does not have a cached OCSP response for the contract certificate. |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EVConnectedPreSession</i> |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Test System sends an AuthorizeRequest With idToken.idToken <Configured valid_idtoken_idtoken> idToken.type <Configured valid_idtoken_type> iso15118CertificateHashData is absent certificate from keystore | 2. The CSMS sends an OCSP request to responder URL of certificate to check validity |
| 3. The Test System OCSP service responds that certificate is valid. | 4. The CSMS responds with a AuthorizeResponse |
| 5. The Test System sends a TransactionEventRequest With triggerReason <i>Authorized</i> | 6. The CSMS responds with a TransactionEventResponse |
| 5. Execute Reusable State <i>EnergyTransferStarted</i> | |

| |
|--|
| Tool validations |
| * Step 2: CSMS sends an OCSP request for certificate * Step 3: Test System checks that received request for certificate is valid AND key type = ECDSA AND certificate chain contains at least one subCA * Step 4: Message: AuthorizeResponse - idTokenInfo.status <i>Accepted</i> - certificateStatus <i>Accepted</i> * Step 6: Message: TransactionEventResponse - idTokenInfo.status <i>Accepted</i> |
| Post scenario validations: N/a |

D Local Authorization List Management

TC_D_01_CSMS: Send Local Authorization List - Full

| | |
|-------------------|---|
| Test case name | Send Local Authorization List - Full |
| Test case Id | TC_D_01_CSMS |
| Use case Id(s) | D01 |
| Requirement(s) | D01_FR_01, D01_FR_06, D01_FR_18 |
| System under test | CSMS |
| Description | The CSMS sends a Local Authorization List which a Charging Station can use for the authorization of idTokens. The list MAY be either a full list to replace the current list in the Charging Station or it MAY be a differential list with updates to be applied to the current list in the Charging Station. |
| Purpose | To verify if the CSMS is able to send a Full Local Authorization List according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2 The Test System responds with a GetLocalListVersionResponse with versionNumber 1 | 1. The CSMS sends a GetLocalListVersionRequest |
| <u>Note(s)</u> : This step is optional | |
| 4 The Test System responds with a SendLocalListResponse with status Accepted | 3. The CSMS sends a SendLocalListRequest |
| <u>Note(s)</u> : If the Local Authorization List is too big for one message, step 1 and 2 will be repeated | |

| |
|---|
| Tool validations |
| * Step 1: Message SendLocalListRequest - updateType <i>Full</i> - versionNumber <i><Bigger than 0></i> - localAuthorizationList <i><Not empty></i> - localAuthorizationList[n].idTokenInfo <i><Not empty></i> |
| Post scenario validations: - N/a |

TC_D_02_CSMS: Send Local Authorization List - Differential Update

| | |
|-------------------|---|
| Test case name | Send Local Authorization List - Differential Update |
| Test case Id | TC_D_02_CSMS |
| Use case Id(s) | D01 |
| Requirement(s) | D01_FR_01, D01_FR_06, D01_FR_18 |
| System under test | CSMS |
| Description | The CSMS sends a Local Authorization List which a Charging Station can use for the authorization of idTokens. The list MAY be either a full list to replace the current list in the Charging Station or it MAY be a differential list with updates to be applied to the current list in the Charging Station. |
| Purpose | To verify if the CSMS is able to send a Differential Local Authorization List according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action:</u> Request the CSMS to send a Local Authorization list to the Charging Station with type Differential and some idTokens in the message | |
| 2 The Test System responds with a GetLocalListVersionResponse with versionNumber 1 | 1. The CSMS sends a GetLocalListVersionRequest |
| 4 The Test System responds with a SendLocalListResponse with status Accepted | 3. The CSMS sends a SendLocalListRequest |
| <u>Note(s):</u> If the Local Authorization List is too big for one message, step 1 and 2 will be repeated | |

| |
|--|
| Tool validations |
| * Step 1: Message SendLocalListRequest - updateType <i>Differential</i> - versionNumber <i><Bigger than currently configured in Test System></i> - localAuthorizationList <i><Not empty></i> |
| Post scenario validations: - N/a |

TC_D_03_CSMS: Send Local Authorization List - Differential Remove

| | |
|-------------------|---|
| Test case name | Send Local Authorization List - Differential Remove |
| Test case Id | TC_D_03_CSMS |
| Use case Id(s) | D01 |
| Requirement(s) | D01_FR_01, D01_FR_06, D01_FR_18, D01_FR_17 |
| System under test | CSMS |
| Description | The CSMS sends a Local Authorization List which a Charging Station can use for the authorization of idTokens. The list MAY be either a full list to replace the current list in the Charging Station or it MAY be a differential list with updates to be applied to the current list in the Charging Station. |
| Purpose | To verify if the CSMS is able to send a Differential Local Authorization List with data without idToken according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action</u> : Request the CSMS to send a Local Authorization list to the Charging Station with type Differential and AuthorizationData elements without idTokenInfo in the message | |
| 2 The Test System responds with a SendLocalListResponse with status Accepted | 1. The CSMS sends a SendLocalListRequest |
| <u>Note(s)</u> : If the Local Authorization List is too big for one message, step 1 and 2 will be repeated | |

| |
|---|
| Tool validations |
| * Step 1: Message SendLocalListRequest - updateType <i>Differential</i> - versionNumber <i><Bigger than currently configured in Test System></i> - localAuthorizationList <i><AuthorizationData elements without idTokenInfo></i> |
| Post scenario validations: - N/a |

TC_D_04_CSMS: Send Local Authorization List - Full with empty list

| | |
|-------------------|---|
| Test case name | Send Local Authorization List - Full with empty list |
| Test case Id | TC_D_04_CSMS |
| Use case Id(s) | D01 |
| Requirement(s) | D01_FR_01, D01_FR_06, D01_FR_18 |
| System under test | CSMS |
| Description | The CSMS sends a Local Authorization List which a Charging Station can use for the authorization of idTokens. The list MAY be either a full list to replace the current list in the Charging Station or it MAY be a differential list with updates to be applied to the current list in the Charging Station. |
| Purpose | To verify if the CSMS is able to send a Full Local Authorization List without data according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action</u> : Request the CSMS to send a Local Authorization list to the Charging Station with type full and without AuthorizationData elements in the message | |
| 2 The Test System responds with a SendLocalListResponse with status Accepted | 1. The CSMS sends a SendLocalListRequest |
| <u>Note(s)</u> : If the Local Authorization List is too big for one message, step 1 and 2 will be repeated | |

| |
|---|
| Tool validations |
| * Step 1: Message SendLocalListRequest - updateType <i>Full</i> - localAuthorizationList <i><Empty></i> |
| Post scenario validations: - N/a |

TC_D_08_CSMS: Get Local List Version - Success

| | |
|-------------------|--|
| Test case name | Get Local List Version - Success |
| Test case Id | TC_D_08_CSMS |
| Use case Id(s) | D02 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | The CSMS can request a Charging Station for the version number of the Local Authorization List by sending a GetLocalListVersionRequest . |
| Purpose | To verify if the CSMS is able to request the Local Authorization List version according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action:</u> Request the CSMS to get a Local Authorization list version | |
| 2 The Test System responds with a GetLocalListVersionResponse with versionNumber <Configured versionNumber> | 1. The CSMS sends a GetLocalListVersionRequest |

| |
|-------------------------------------|
| Tool validations |
| - N/a |
| Post scenario validations: - N/a |

TC_D_09_CSMS: Get Local List Version - No list available

| | |
|--------------------------|--|
| Test case name | Get Local List Version - No list available |
| Test case Id | TC_D_09_CSMS |
| Use case Id(s) | D02 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | The CSMS can request a Charging Station for the version number of the Local Authorization List by sending a GetLocalListVersionRequest . |
| Purpose | To verify if the CSMS is able to request the Local Authorization List version according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | N/a |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action:</u> Request the CSMS to get a Local Authorization list version | |
| 2 The Test System responds with a GetLocalListVersionResponse with versionNumber 0 | 1. The CSMS sends a GetLocalListVersionRequest |

| |
|--|
| Tool validations |
| - N/a |
| Post scenario validations: - N/a |

E Transactions

TC_E_01_CSMS: Start transaction options - PowerPathClosed

| | |
|-------------------|--|
| Test case name | Start transaction options - PowerPathClosed |
| Test case Id | TC_E_01_CSMS |
| Use case Id(s) | E01(S5) |
| Requirement(s) | E01.FR.05 |
| System under test | CSMS |
| Description | OCPP 2.x.x has a flexible transaction mechanism that allows the transaction start and stop points to be configured differently. This test covers one of the start options. |
| Purpose | To verify if the CSMS is able to handle a Charging Station that starts a transaction when the power path has been closed. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Test System sends an AuthorizeRequest With idToken.idToken <Configured valid_idtoken_idtoken> idToken.type <Configured valid_idtoken_type> | 2. The CSMS responds with an AuthorizeResponse |
| 3. The Test System notifies the CSMS about the status change of the connector. Message: StatusNotificationRequest - connectorStatus is <i>Occupied</i> Message: NotifyEventRequest - trigger is <i>Delta</i> - actualValue is <i>Occupied</i> - component.name is <i>Connector</i> - variable.name is <i>AvailabilityState</i> | 4. The CSMS responds accordingly. |
| 5. The Test System sends a TransactionEventRequest With eventType is <i>Started</i> triggerReason is <i>ChargingStateChanged</i> idToken.idToken <Configured valid_idtoken_idtoken> idToken.type <Configured valid_idtoken_type> evse.id is <Configured evseld> evse.connectorId is <Configured connectorId> transactionInfo.chargingState is <i>SuspendedEVSE</i> | 6. The CSMS responds with a TransactionEventResponse |
| 7. The Test System sends a TransactionEventRequest With eventType is <i>Updated</i> triggerReason is <i>ChargingStateChanged</i> transactionInfo.chargingState is <i>Charging</i> | 8. The CSMS responds with a TransactionEventResponse |

| Tool validations |
|--|
| <p>* Step 2:</p> <p>Message: AuthorizeResponse</p> <p>- idTokenInfo.status must be <i>Accepted</i></p> <p>* Step 6:</p> <p>Message: TransactionEventResponse</p> <p>- idTokenInfo.status must be <i>Accepted</i></p> |
| <p>Post scenario validations:</p> <p>N/a</p> |

TC_E_02_CSMS: Start transaction options - EnergyTransfer

| | |
|-------------------|--|
| Test case name | Start transaction options - EnergyTransfer |
| Test case Id | TC_E_02_CSMS |
| Use case Id(s) | E01(S6) |
| Requirement(s) | E01.FR.06 |
| System under test | CSMS |
| Description | OCPP 2.x.x has a flexible transaction mechanism that allows the transaction start and stop points to be configured differently. This test covers one of the start options. |
| Purpose | To verify if the CSMS is able to handle a Charging Station that starts a transaction when the energy transfer starts. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Test System sends an AuthorizeRequest With idToken.idToken <Configured valid_idtoken_idtoken> idToken.type <Configured valid_idtoken_type> | 2. The CSMS responds with an AuthorizeResponse |
| 3. The Test System notifies the CSMS about the status change of the connector. Message: StatusNotificationRequest - connectorStatus is <i>Occupied</i> Message: NotifyEventRequest - trigger is <i>Delta</i> - actualValue is <i>Occupied</i> - component.name is <i>Connector</i> - variable.name is <i>AvailabilityState</i> | 4. The CSMS responds accordingly. |
| 5. The Test System sends a TransactionEventRequest With eventType is <i>Started</i> triggerReason is <i>ChargingStateChanged</i> idToken.idToken <Configured valid_idtoken_idtoken> idToken.type <Configured valid_idtoken_type> evse.id is <Configured evseld> evse.connectorId is <Configured connectorId> transactionInfo.chargingState is <i>Charging</i> | 6. The CSMS responds with a TransactionEventResponse |

| |
|---|
| Tool validations |
| * Step 2: Message: AuthorizeResponse - idTokenInfo.status must be <i>Accepted</i> * Step 6: Message: TransactionEventResponse - idTokenInfo.status must be <i>Accepted</i> |

| |
|--|
| Tool validations |
| Post scenario validations: N/a |

TC_E_03_CSMS: Local start transaction - Cable plugin first - Success

| | |
|-------------------|--|
| Test case name | Local start transaction - Cable plugin first - Success |
| Test case Id | TC_E_03_CSMS |
| Use case Id(s) | E02 |
| Requirement(s) | E02.FR.02 |
| System under test | CSMS |
| Description | OCPP 2.x.x allows an EV driver to either first connect the EV and EVSE OR present a form of identification. Both sequences will result in being able to charge. |
| Purpose | To verify if the CSMS is able to handle a Charging Station that is able to start a charging session when the EV driver first connects the EV and EVSE, before authorization. |
| Prerequisite(s) | N/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EVConnectedPreSession</i> |

| | |
|--|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>Authorized</i> | |
| 2. Execute Reusable State <i>EnergyTransferStarted</i> | |

| |
|-----------------------------------|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_E_04_CSMS: Local start transaction - Authorization first - Success

| | |
|-------------------|--|
| Test case name | Local start transaction - Authorization first - Success |
| Test case Id | TC_E_04_CSMS |
| Use case Id(s) | E03 |
| Requirement(s) | E03.FR.02 |
| System under test | CSMS |
| Description | OCPP 2.x.x allows an EV driver to either first connect the EV and EVSE OR present a form of identification. Both sequences will result in being able to charge. |
| Purpose | To verify if the CSMS is able to handle a Charging Station that is able to start a charging session when the EV driver first presents a form of identification, before connecting the EV and EVSE. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>Authorized</i> | |
| 2. Execute Reusable State <i>EnergyTransferStarted</i> | |

| |
|-----------------------------------|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_E_39_CSMS: Stop transaction options - Deauthorized - timeout

| | |
|-------------------|---|
| Test case name | Stop transaction options - Deauthorized - timeout |
| Test case Id | TC_E_39_CSMS |
| Use case Id(s) | E03, E06 |
| Requirement(s) | E03.FR.04, E03.FR.05, E06.FR.04 |
| System under test | CSMS |
| Description | OCPP 2.x.x allows an EV driver to either first connect the EV and EVSE OR present a form of identification. Both sequences will result in being able to charge. |
| Purpose | To verify if the CSMS is able to handle a Charging Station that deauthorizes the transaction after the EVConnectionTimeout has expired. |
| Prerequisite(s) | N/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>Authorized</i> |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <p>1. The Test System sends a TransactionEventRequest With triggerReason is <i>EVConnectTimeout</i> transactionInfo.stoppedReason is <i>Timeout</i> eventType is <i>Ended</i></p> <p><u>Note(s):</u> - This step will be executed after the _<Configured EV connection timeout> expires._</p> | <p>2. The CSMS responds with a TransactionEventResponse</p> |

| |
|-----------------------------------|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_E_14_CSMS: Stop transaction options - EVDisconnected - Charging Station side

| | |
|--------------------------|---|
| Test case name | Stop transaction options - EVDisconnected - Charging Station side |
| Test case Id | TC_E_14_CSMS |
| Use case Id(s) | E06(S2) |
| Requirement(s) | E06.FR.02 |
| System under test | CSMS |
| Description | OCPP 2.x.x has a flexible transaction mechanism that allows the transaction start and stop points to be configured differently. This test covers one of the stop options. |
| Purpose | To verify if the CSMS is able to handle a Charging Station that stops a transaction when the EV and EVSE are disconnected at the Charging Station side. |
| Prerequisite(s) | N/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EVConnectedPostSession</i> |

| | |
|---|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>EVD</i> Disconnected | |

| |
|--|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_E_20_CSMS: Stop transaction options - EVDisconnected - EV side (able to charge IEC 61851-1 EV)

| | |
|-------------------|---|
| Test case name | Stop transaction options - EVDisconnected - EV side (able to charge IEC 61851-1 EV) |
| Test case Id | TC_E_20_CSMS |
| Use case Id(s) | E06(S2), E10 |
| Requirement(s) | E06.FR.02 |
| System under test | CSMS |
| Description | OCPP 2.x.x has a flexible transaction mechanism that allows the transaction start and stop points to be configured differently. This test covers one of the stop options. |
| Purpose | To verify if the CSMS is able to handle a Charging Station that stops a transaction when the EV and EVSE are disconnected at the EV side. |
| Prerequisite(s) | N/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferSuspended</i> |

| | |
|---|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>EVDIsconnected</i> | |

| |
|-----------------------------------|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_E_15_CSMS: Stop transaction options - StopAuthorized - Local

| | |
|-------------------|---|
| Test case name | Stop transaction options - StopAuthorized - Local |
| Test case Id | TC_E_15_CSMS |
| Use case Id(s) | E06(S3) |
| Requirement(s) | E06.FR.03 |
| System under test | CSMS |
| Description | OCPP 2.x.x has a flexible transaction mechanism that allows the transaction start and stop points to be configured differently. This test covers one of the stop options. |
| Purpose | To verify if the CSMS is able to handle a Charging Station that stops a transaction when the EV driver locally stops the transaction. |
| Prerequisite(s) | N/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> |

| Main (Test scenario) | |
|---|---|
| Charging Station | CSMS |
| 1. The Test System sends a TransactionEventRequest With triggerReason is <i>StopAuthorized</i> transactionInfo.stoppedReason is <i>Local</i> eventType is <i>Ended</i> | 2. The CSMS responds with a TransactionEventResponse |

| |
|-----------------------------------|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_E_21_CSMS: Stop transaction options - StopAuthorized - Remote

| | |
|-------------------|---|
| Test case name | Stop transaction options - StopAuthorized - Remote |
| Test case Id | TC_E_21_CSMS |
| Use case Id(s) | E06(S3) AND F03 |
| Requirement(s) | E06.FR.03,F03.FR.01,F03.FR.09, F03.FR.10 |
| System under test | CSMS |
| Description | OCPP 2.x.x has a flexible transaction mechanism that allows the transaction start and stop points to be configured differently. This test covers one of the stop options. |
| Purpose | To verify if the CSMS is able to handle a Charging Station that stops a transaction when it receives a RequestStopTransactionRequest. |
| Prerequisite(s) | N/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> |

| Main (Test scenario) | |
|---|---|
| Charging Station | CSMS |
| <u>Manual Action</u> : Trigger the CSMS to request the Charging Station to stop the ongoing transaction. | |
| 2. The Test System responds with a RequestStopTransactionResponse with status <i>Accepted</i> | 1. The CSMS sends a RequestStopTransactionRequest |
| 3. The Test System sends a TransactionEventRequest . with triggerReason is <i>RemoteStop</i> transactionInfo.stoppedReason is <i>Remote</i> eventType is <i>Ended</i> | 4. The CSMS responds with a TransactionEventResponse . |

| |
|--|
| Tool validations |
| * Step 1: Message: RequestStopTransactionRequest - transactionId must equal <transactionId provided by the Test System in before state.> |
| Post scenario validations: N/a |

TC_E_09_CSMS: Start transaction options - EVConnected

| | |
|-------------------|--|
| Test case name | Start transaction options - EVConnected |
| Test case Id | TC_E_09_CSMS |
| Use case Id(s) | E01(S2) |
| Requirement(s) | E01.FR.02 |
| System under test | CSMS |
| Description | OCPP 2.x.x has a flexible transaction mechanism that allows the transaction start and stop points to be configured differently. This test covers one of the start options. |
| Purpose | To verify if the CSMS is able to handle a Charging Station that starts a transaction when the EV and EVSE are connected. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <p>1. The Test System notifies the CSMS about the status change of the connector.</p> <p>Message: StatusNotificationRequest</p> <ul style="list-style-type: none"> - connectorStatus is <i>Occupied</i> <p>Message: NotifyEventRequest</p> <ul style="list-style-type: none"> - trigger is <i>Delta</i> - actualValue is <i>Occupied</i> - component.name is <i>Connector</i> - variable.name is <i>AvailabilityState</i> | <p>2. The CSMS responds accordingly.</p> |
| <p>3. The Test System sends a TransactionEventRequest</p> <p>With eventType is <i>Started</i></p> <p>triggerReason is <i>CablePluggedIn</i></p> <p>evse.id is <i><Configured evseld></i></p> <p>evse.connectorId is <i><Configured connectorId></i></p> <p>transactionInfo.chargingState is <i>EVConnected</i></p> | <p>4. The CSMS responds with a TransactionEventResponse</p> |

| |
|-----------------------------------|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_E_10_CSMS: Start transaction options - Authorized - Local

| | |
|-------------------|--|
| Test case name | Start transaction options - Authorized - Local |
| Test case Id | TC_E_10_CSMS |
| Use case Id(s) | E01(S3) |
| Requirement(s) | E01.FR.03 |
| System under test | CSMS |
| Description | OCPP 2.x.x has a flexible transaction mechanism that allows the transaction start and stop points to be configured differently. This test covers one of the start options. |
| Purpose | To verify if the CSMS is able to handle a Charging Station that starts a transaction when the EV and EVSE are connected. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|---|---|
| Charging Station | CSMS |
| 1. The Test System sends an AuthorizeRequest With idToken.idToken <Configured valid_idtoken_idtoken> idToken.type <Configured valid_idtoken_type> | 2. The CSMS responds with an AuthorizeResponse |
| 3. The Test System sends a TransactionEventRequest With eventType is Started triggerReason is Authorized idToken.idToken <Configured valid_idtoken_idtoken> idToken.type <Configured valid_idtoken_type> | 4. The CSMS responds with a TransactionEventResponse |

| |
|--|
| Tool validations |
| * Step 2: Message: AuthorizeResponse - idTokenInfo.status must be <i>Accepted</i> |
| * Step 4: Message: TransactionEventResponse - idTokenInfo.status must be <i>Accepted</i> |
| Post scenario validations: N/a |

TC_E_11_CSMS: Start transaction options - DataSigned

| | |
|-------------------|--|
| Test case name | Start transaction options - DataSigned |
| Test case Id | TC_E_11_CSMS |
| Use case Id(s) | E01(S4) |
| Requirement(s) | E01.FR.04 |
| System under test | CSMS |
| Description | OCPP 2.x.x has a flexible transaction mechanism that allows the transaction start and stop points to be configured differently. This test covers one of the start options. |
| Purpose | To verify if the CSMS is able to handle a Charging Station that starts a transaction when the signed meter values are received. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Test System sends an AuthorizeRequest With idToken.idToken <Configured valid_idtoken_idtoken> idToken.type <Configured valid_idtoken_type> | 2. The CSMS responds with an AuthorizeResponse |
| 3. The Test System notifies the CSMS about the status change of the connector. Message: StatusNotificationRequest - connectorStatus is <i>Occupied</i> Message: NotifyEventRequest - trigger is <i>Delta</i> - actualValue is <i>Occupied</i> - component.name is <i>Connector</i> - variable.name is <i>AvailabilityState</i> | 4. The CSMS responds accordingly. |
| 5. The Test System sends a TransactionEventRequest With eventType is <i>Started</i> triggerReason is <i>SignedDataReceived</i> idToken.idToken <Configured valid_idtoken_idtoken> idToken.type <Configured valid_idtoken_type> evse.id is <Configured evseld> evse.connectorId is <Configured connectorId> meterValue is provided with the following values: sampledValue.value is <i>0.0</i> sampledValue.context is <i>Transaction.Begin</i> sampledValue.signedMeterValue is <Generated <i>SignedMeterValueType</i> > | 6. The CSMS responds with a TransactionEventResponse |
| 7. The Test System sends a TransactionEventRequest With eventType is <i>Updated</i> triggerReason is <i>ChargingStateChanged</i> transactionInfo.chargingState is <i>Charging</i> | 8. The CSMS responds with a TransactionEventResponse |

| Tool validations |
|--|
| <p>* Step 2:</p> <p>Message: AuthorizeResponse</p> <p>- idTokenInfo.status must be <i>Accepted</i></p> <p>* Step 6:</p> <p>Message: TransactionEventResponse</p> <p>- idTokenInfo.status must be <i>Accepted</i></p> |
| <p>Post scenario validations:</p> <p>N/a</p> |

TC_E_12_CSMS: Start transaction options - ParkingBayOccupied

| | |
|--------------------------|--|
| Test case name | Start transaction options - ParkingBayOccupied |
| Test case Id | TC_E_12_CSMS |
| Use case Id(s) | E01(S1) |
| Requirement(s) | E01.FR.01 |
| System under test | CSMS |
| Description | OCPP 2.x.x has a flexible transaction mechanism that allows the transaction start and stop points to be configured differently. This test covers one of the start options. |
| Purpose | To verify if the CSMS is able to handle a Charging Station that starts a transaction when the EV and EVSE are connected. |
| Prerequisite(s) | N/a |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Test System sends a TransactionEventRequest With eventType is <i>Started</i> triggerReason is <i>EVDetected</i> | 2. The CSMS responds with a TransactionEventResponse |

| |
|--|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_E_38_CSMS: Local start transaction - EV not ready

| | |
|-------------------|---|
| Test case name | Local start transaction - EV not ready |
| Test case Id | TC_E_38_CSMS |
| Use case Id(s) | E03 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | OCPP 2.x.x allows an EV driver to either first connect the EV and EVSE OR present a form of identification. Both sequences will result in being able to charge. |
| Purpose | To verify if the CSMS is able to handle a Charging Station that reports an EV is not ready to start the energy transfer (yet). |
| Prerequisite(s) | N/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>Authorized</i> |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>EVConnectedPreSession</i> | |
| 2. The Test System sends a TransactionEventRequest With triggerReason is <i>ChargingStateChanged</i> transactionInfo.chargingState is <i>SuspendedEV</i> eventType is <i>Updated</i> | 3. The CSMS responds with a TransactionEventResponse |

| |
|-----------------------------------|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_E_07_CSMS: Stop transaction options - PowerPathClosed - Local stop

| | |
|--------------------------|---|
| Test case name | Stop transaction options - PowerPathClosed - Local stop |
| Test case Id | TC_E_07_CSMS |
| Use case Id(s) | E06(S5) |
| Requirement(s) | E06.FR.06 |
| System under test | CSMS |
| Description | OCPP 2.x.x has a flexible transaction mechanism that allows the transaction start and stop points to be configured differently. This test covers one of the stop options. |
| Purpose | To verify if the CSMS is able to handle a Charging Station that stops a transaction when it is locally stopped by an EV driver. |
| Prerequisite(s) | N/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Test System sends a TransactionEventRequest With triggerReason is <i>StopAuthorized</i> transactionInfo.stoppedReason is <i>Local</i> eventType is <i>Ended</i> | 2. The CSMS responds with a TransactionEventResponse |

| |
|--|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_E_08_CSMS: Stop transaction options - EnergyTransfer stopped - StopAuthorized

| | |
|-------------------|---|
| Test case name | Stop transaction options - EnergyTransfer stopped - StopAuthorized |
| Test case Id | TC_E_08_CSMS |
| Use case Id(s) | E06(S6) |
| Requirement(s) | E06.FR.07 |
| System under test | CSMS |
| Description | OCPP 2.x.x has a flexible transaction mechanism that allows the transaction start and stop points to be configured differently. This test covers one of the stop options. |
| Purpose | To verify if the CSMS is able to handle a Charging Station that stops a transaction when the energy transfer stopped normally. |
| Prerequisite(s) | N/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>StopAuthorized</i> |

| Main (Test scenario) | |
|---|---|
| Charging Station | CSMS |
| 1. The Test System sends a TransactionEventRequest With triggerReason is <i>ChargingStateChanged</i> transactionInfo.chargingState is <i>EVConnected</i> transactionInfo.stoppedReason is <i>Local</i> eventType is <i>Ended</i> | 2. The CSMS responds with a TransactionEventResponse |

| |
|-----------------------------------|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_E_16_CSMS: Stop transaction options - Deauthorized - Invalid idToken

| | |
|--------------------------|--|
| Test case name | Stop transaction options - Deauthorized - Invalid idToken |
| Test case Id | TC_E_16_CSMS |
| Use case Id(s) | E06(S3) |
| Requirement(s) | E06.FR.04,E01.FR.11,E01.FR.12 |
| System under test | CSMS |
| Description | OCPP 2.x.x has a flexible transaction mechanism that allows the transaction start and stop points to be configured differently. This test covers one of the start options. |
| Purpose | To verify if the CSMS is able to handle a Charging Station that stops a transaction when the transaction gets deauthorized by the status from the idTokenInfo at a TransactionEventResponse message and it has been configured to do so. |
| Prerequisite(s) | N/a |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Test System sends a TransactionEventRequest With triggerReason is <i>Authorized</i> idToken.idToken <Configured invalid_idtoken_idtoken> idToken.type <Configured invalid_idtoken_type> eventType is <i>Started</i> | 2. The CSMS responds with a TransactionEventResponse |
| 3. The Test System sends a TransactionEventRequest With eventType <i>Ended</i> triggerReason <i>Deauthorized</i> transactionInfo.stoppedReason <i>DeAuthorized</i> | 4. The CSMS responds with a TransactionEventResponse |

| |
|---|
| Tool validations |
| * Step 2: Message: TransactionEventResponse - idTokenInfo.status must be <i>Invalid</i> or <i>Unknown</i> + |
| Post scenario validations: N/a |

TC_E_17_CSMS: Stop transaction options - Deauthorized - EV side disconnect

| | |
|-------------------|---|
| Test case name | Stop transaction options - Deauthorized - EV side disconnect |
| Test case Id | TC_E_17_CSMS |
| Use case Id(s) | E06(S3) |
| Requirement(s) | E06.FR.04 |
| System under test | CSMS |
| Description | OCPP 2.x.x has a flexible transaction mechanism that allows the transaction start and stop points to be configured differently. This test covers one of the start options. |
| Purpose | To verify if the CSMS is able to handle a Charging Station that stops a transaction when the transaction gets deauthorized by a connection loss from the EV side and it has been configured to do so. |
| Prerequisite(s) | N/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferSuspended</i> |

| Main (Test scenario) | |
|---|---|
| Charging Station | CSMS |
| 1. The Test System sends a TransactionEventRequest triggerReason must be <i>EVCommunicationLost</i> transactionInfo.chargingState must be <i>Idle</i> transactionInfo.stoppedReason must be <i>EVDisconnected</i> eventType must be <i>Ended</i> | 2. The CSMS responds with a TransactionEventResponse |

| |
|-----------------------------------|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_E_22_CSMS: Stop transaction options - EnergyTransfer stopped - SuspendedEV

| | |
|-------------------|---|
| Test case name | Stop transaction options - EnergyTransfer stopped - SuspendedEV |
| Test case Id | TC_E_22_CSMS |
| Use case Id(s) | E06(S6) |
| Requirement(s) | E06.FR.07 |
| System under test | CSMS |
| Description | OCPP 2.x.x has a flexible transaction mechanism that allows the transaction start and stop points to be configured differently. This test covers one of the stop options. |
| Purpose | To verify if the CSMS is able to handle a Charging Station that stops a transaction when the energy transfer stopped by the EV. |
| Prerequisite(s) | N/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> |

| Main (Test scenario) | |
|---|---|
| Charging Station | CSMS |
| 1. The Test System sends a TransactionEventRequest With triggerReason is <i>ChargingStateChanged</i> transactionInfo.chargingState is <i>SuspendedEV</i> transactionInfo.stoppedReason is <i>StoppedByEV</i> eventType is <i>Ended</i> | 2. The CSMS responds with a TransactionEventResponse |

| |
|-----------------------------------|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_E_19_CSMS: Stop transaction options - ParkingBayUnoccupied

| | |
|--------------------------|---|
| Test case name | Stop transaction options - ParkingBayUnoccupied |
| Test case Id | TC_E_19_CSMS |
| Use case Id(s) | E06(S1) |
| Requirement(s) | E06.FR.01 |
| System under test | CSMS |
| Description | OCPP 2.x.x has a flexible transaction mechanism that allows the transaction start and stop points to be configured differently. This test covers one of the stop options. |
| Purpose | To verify if the CSMS is able to handle a Charging Station that stops a transaction when the EV left the parking bay. |
| Prerequisite(s) | N/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EVDisconnected</i> |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Test System sends a TransactionEventRequest With triggerReason is <i>EVDeparted</i> transactionInfo.stoppedReason is <i>Local</i> eventType is <i>Ended</i> | 2. The CSMS responds with a TransactionEventResponse |

| |
|--|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_E_26_CSMS: Disconnect cable on EV-side - Suspend transaction

| | |
|-------------------|--|
| Test case name | Disconnect cable on EV-side - Suspend transaction |
| Test case Id | TC_E_26_CSMS |
| Use case Id(s) | E10 |
| Requirement(s) | E10.FR.01 |
| System under test | CSMS |
| Description | The Charging Station can behave in several different ways when the cable is disconnected at the EV side, based on its configuration. This test case tests one of the possible configuration settings. |
| Purpose | To verify if the CSMS can handle a Charging Station that suspends the transaction when the EV and EVSE are disconnected at the EV side AND is able restart the energy transfer after reconnecting the EV and EVSE. |
| Prerequisite(s) | N/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferSuspended</i> |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Test System sends a TransactionEventRequest With triggerReason is <i>EVCommunicationLost</i> transactionInfo.chargingState is <i>Idle</i> eventType is <i>Updated</i> | 2. The CSMS responds with a TransactionEventResponse |
| 3. The Test System notifies the CSMS about the current state of the connector. Message: StatusNotificationRequest - connectorStatus <i>Available</i> - evseld <i><Configured evseld></i> - connectorId <i><Configured connectorId></i> Message: NotifyEventRequest - trigger <i>Delta</i> - actualValue <i>"Available"</i> - component.name <i>"Connector"</i> - component.evse.id <i><Configured evseld></i> - component.evse.connectorId <i><Configured connectorId></i> - variable.name <i>"AvailabilityState"</i> | 4. The CSMS responds accordingly. |
| 5. The Test System sends a TransactionEventRequest With triggerReason is <i>CablePluggedIn</i> transactionInfo.chargingState is <i>EVConnected</i> eventType is <i>Updated</i> | 6. The CSMS responds with a TransactionEventResponse |
| 7. The Test System sends a TransactionEventRequest With triggerReason is <i>ChargingStateChanged</i> transactionInfo.chargingState is <i>Charging</i> eventType is <i>Updated</i> | 8. The CSMS responds with a TransactionEventResponse |

| |
|-----------------------------------|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_E_29_CSMS: Check Transaction status - Transaction with id ongoing - with message in queue

| | |
|-------------------|--|
| Test case name | Check Transaction status - Transaction with id ongoing - with message in queue |
| Test case Id | TC_E_29_CSMS |
| Use case Id(s) | E14 |
| Requirement(s) | E14.FR.02,E14.FR.04 |
| System under test | CSMS |
| Description | The CSMS is able to request the status of a transaction and to find out whether there are queued transaction-related messages, using the GetTransactionStatusRequest message. |
| Purpose | To verify if the CSMS is able to request the status of queued TransactionEventRequest messages from a specific transaction by sending a GetTransactionStatusRequest with a transactionId. The Test System will respond that there are message(s) queued belonging to the ongoing transaction with the requested id. |
| Prerequisite(s) | N/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> |

| Main (Test scenario) | |
|--|---|
| Charging Station | CSMS |
| 1. The Test System closes the WebSocket connection. | |
| 2. The Test System waits a number of seconds equal to <i>_<Configured Transaction Duration></i> , then it will reconnect to the CSMS._ | |
| 4. The Test System responds with a GetTransactionStatusResponse With ongoingIndicator is <i>true</i> messagesInQueue is <i>true</i> | 3. The CSMS sends a GetTransactionStatusRequest |
| 5. The Test System sends a TransactionEventRequest With eventType is <i>Updated</i> meterValues is present. offline is <i>true</i> | 6. The CSMS responds with a TransactionEventResponse |

| |
|---|
| Tool validations |
| * Step 3: Message: GetTransactionStatusRequest - transactionId <i><Generated transactionId from Before></i> |
| Post scenario validations: N/a |

TC_E_30_CSMS: Check Transaction status - Transaction with id ongoing - without message in queue

| | |
|-------------------|---|
| Test case name | Check Transaction status - Transaction with id ongoing - without message in queue |
| Test case Id | TC_E_30_CSMS |
| Use case Id(s) | E14 |
| Requirement(s) | E14.FR.02,E14.FR.05 |
| System under test | CSMS |
| Description | The CSMS is able to request the status of a transaction and to find out whether there are queued transaction-related messages, using the GetTransactionStatusRequest message. |
| Purpose | To verify if the CSMS is able to request the status of queued TransactionEventRequest messages from a specific transaction by sending a GetTransactionStatusRequest with a transactionId. The Test System will respond that there is NO message queued belonging to the ongoing transaction with the requested id. |
| Prerequisite(s) | N/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a GetTransactionStatusResponse With ongoingIndicator is <i>true</i> messagesInQueue is <i>false</i> | 1. The CSMS sends a GetTransactionStatusRequest |

| |
|--|
| Tool validations |
| * Step 1: Message: GetTransactionStatusRequest - transactionId must be <Generated transactionId from Before> |
| Post scenario validations: N/a |

TC_E_31_CSMS: Check Transaction status - Transaction with id ended - with message in queue

| | |
|-------------------|---|
| Test case name | Check Transaction status - Transaction with id ended - with message in queue |
| Test case Id | TC_E_31_CSMS |
| Use case Id(s) | E14 |
| Requirement(s) | E14.FR.03,E14.FR.04 |
| System under test | CSMS |
| Description | The CSMS is able to request the status of a transaction and to find out whether there are queued transaction-related messages, using the GetTransactionStatusRequest message. |
| Purpose | To verify if the CSMS is able to request the status of queued TransactionEventRequest messages from a specific transaction by sending a GetTransactionStatusRequest with a transactionId. The Test System will respond that there are message(s) queued belonging to an ended transaction with the requested id. |
| Prerequisite(s) | N/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Test System closes the WebSocket connection. | |
| 2. The Test System waits a number of seconds equal to <i><Configured Transaction duration></i> , then it will reconnect to the CSMS. | |
| 3. The Test System sends a StatusNotificationRequest With evseld is <i><Configured evseld></i> connectorId is <i><Configured connectorId></i> connectorStatus is <i>Available</i> | 4. The CSMS responds with a StatusNotificationResponse |
| 5. The Test System sends a TransactionEventRequest With eventType is <i>Ended</i> offline is <i>true</i> triggerReason is <i>EVCommunicationLost</i> transactionInfo.chargingState is <i>Idle</i> seqNo <i><Skips two sequence number values></i> | 6. The CSMS responds with a TransactionEventResponse |
| 8. The Test System responds with a GetTransactionStatusResponse With ongoingIndicator is <i>false</i> messagesInQueue is <i>true</i> | 7. The CSMS sends a GetTransactionStatusRequest |
| 9. The Test System sends a TransactionEventRequest With triggerReason is <i>StopAuthorized</i> eventType is <i>Updated</i> offline is <i>true</i> seqNo <i><This is the first of the two skipped values></i> | 10. The CSMS responds with a TransactionEventResponse |
| 11. The Test System sends a TransactionEventRequest With triggerReason is <i>ChargingStateChanged</i> transactionInfo.chargingState is <i>EVConnected</i> eventType is <i>Updated</i> offline is <i>true</i> seqNo <i><This is the second of the two skipped values></i> | 12. The CSMS responds with a TransactionEventResponse |

| |
|--|
| Tool validations |
| * Step 5: Message: GetTransactionStatusRequest - transactionId <Generated transactionId from Before> |
| Post scenario validations: N/a |

TC_E_33_CSMS: Check Transaction status - Without transactionId - with message in queue

| | |
|--------------------------|--|
| Test case name | Check Transaction status - Without transactionId - with message in queue |
| Test case Id | TC_E_33_CSMS |
| Use case Id(s) | E14 |
| Requirement(s) | E14.FR.06,E14.FR.07 |
| System under test | CSMS |
| Description | The CSMS is able to request the status of a transaction and to find out whether there are queued transaction-related messages, using the GetTransactionStatusRequest message. |
| Purpose | To verify if the CSMS is able to request the status of queued TransactionEventRequest messages by sending a GetTransactionStatusRequest without a transactionId. The Test System will respond that there are message(s) queued. |
| Prerequisite(s) | N/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Test System closes the WebSocket connection. | |
| 2. The Test System waits a number of seconds equal to <i><Configured Transaction Duration></i> , then it will reconnect to the CSMS. | |
| 4. The Test System responds with a GetTransactionStatusResponse With ongoingIndicator is omitted. messagesInQueue is <i>true</i> | 3. The CSMS sends a GetTransactionStatusRequest |
| 5. The Test System sends a TransactionEventRequest With eventType is <i>Updated</i> meterValues is present. offline is <i>true</i> | 6. The CSMS responds with a TransactionEventResponse |

| |
|---|
| Tool validations |
| * Step 3: Message: GetTransactionStatusRequest - transactionId must be omitted. |
| Post scenario validations: N/a |

TC_E_34_CSMS: Check Transaction status - Without transactionId - without message in queue

| | |
|-------------------|---|
| Test case name | Check Transaction status - Without transactionId - without message in queue |
| Test case Id | TC_E_34_CSMS |
| Use case Id(s) | E14 |
| Requirement(s) | E14.FR.06,E14.FR.08 |
| System under test | CSMS |
| Description | The CSMS is able to request the status of a transaction and to find out whether there are queued transaction-related messages, using the GetTransactionStatusRequest message. |
| Purpose | To verify if the CSMS is able to request the status of queued TransactionEventRequest messages by sending a GetTransactionStatusRequest without a transactionId. The Test System will respond that there are NO message(s) queued. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a GetTransactionStatusResponse With ongoingIndicator is omitted. messagesInQueue is <i>false</i> | 1. The CSMS sends a GetTransactionStatusRequest |

| |
|---|
| Tool validations |
| * Step 1: Message: GetTransactionStatusRequest - transactionId must be omitted. |
| Post scenario validations: N/a |

TC_E_53_CSMS: Reset Sequence Number - CSMS accepting seqNo = 0 at start of transaction

| | |
|-------------------|--|
| Test case name | Reset Sequence Number - CSMS accepting seqNo = 0 at start of transaction |
| Test case Id | TC_E_53_CSMS |
| Use case Id(s) | E01 |
| Requirement(s) | E01.FR.07 |
| System under test | CSMS |
| Description | OCPP 2.0.1 Edition 2 recommends that seqNo starts at 0 for every transaction. CSMS must therefore be robust to a seqNo that is not continuously increasing, but that restarts for new transactions. Since a TransactionEventRequest cannot be rejected, this can only be detected by either the complete absence of a TransactionEventResponse from CSMS or an otherwise misbehaving CSMS. |
| Purpose | To verify if the CSMS accepts that a new transactions starts with a seqNo = 0. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State EnergyTransferStarted <u>Note(s)</u> : New transaction will use seqNo 0 for the first TransactionEventRequest. | |
| 2. Execute Reusable State EVDIsconnected | |
| 3. Execute Reusable State EnergyTransferStarted <u>Note(s)</u> : New transaction will use seqNo 0 for the first TransactionEventRequest. | |
| 4. Execute Reusable State EVDIsconnected | |

| |
|---|
| Tool validations |
| * Step 1: CSMS accepts the message TransactionEventRequest with eventType = Started and seqNo = 0 and answers with a TransactionEventResponse message. |
| * Step 3: CSMS accepts the message TransactionEventRequest with eventType = Started and seqNo = 0 and answers with a TransactionEventResponse message. |
| Post scenario validations: N/a |

F Remote Control

TC_F_01_CSMS: Remote start transaction - Cable plugin first

| | |
|-------------------|--|
| Test case name | Remote start transaction - Cable plugin first |
| Test case Id | TC_F_01_CSMS |
| Use case Id(s) | F01 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | OCPP 2.x.x allows an EV driver to either first connect the EV and EVSE OR wait for/trigger a RequestStartTransactionRequest. Both sequences will result in being able to charge. |
| Purpose | To verify if the CSMS is able to handle a Charging Station that is starts a charging session when the EV driver first connects the EV and EVSE, before receiving a RequestStartTransactionRequest message. |
| Prerequisite(s) | N/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EVConnectedPreSession</i> |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action:</u> Trigger the CSMS to request the Charging Station to start a transaction. | |
| 2. The Test System responds with a RequestStartTransactionResponse with status <i>Accepted</i> transactionId is <i><Generated transactionId></i> | 1. The CSMS sends a RequestStartTransactionRequest |
| 3. The Test System sends a TransactionEventRequest . with triggerReason is <i>RemoteStart</i> transactionInfo.remoteStartId is <i><By CSMS provided remoteStartId></i> eventType is <i>Updated</i> | 4. The CSMS responds with a TransactionEventResponse . |
| 5. Execute Reusable State <i>EnergyTransferStarted</i> (State is <i>Authorized</i> and <i>_EVConnected</i> = true) | |

| |
|---|
| Tool validations |
| * Step 1: Message: RequestStartTransactionRequest - idToken.idToken <i><Configured valid_idtoken_idtoken></i> - idToken.type <i><Configured valid_idtoken_type></i> |
| Post scenario validations: N/a |

TC_F_02_CSMS: Remote start transaction - Remote start first - AuthorizeRemoteStart is true

| | |
|-------------------|---|
| Test case name | Remote start transaction - Remote start first - AuthorizeRemoteStart is true |
| Test case Id | TC_F_02_CSMS |
| Use case Id(s) | F02 |
| Requirement(s) | F02.FR.01, F01.FR.01 |
| System under test | CSMS |
| Description | OCPP 2.x.x allows an EV driver to either first wait for/trigger a RequestStartTransactionRequest OR connect the EV and EVSE. Both sequences will result in being able to charge. |
| Purpose | To verify if the CSMS is able to handle a Charging Station that starts a charging session when the Charging Stations receives a RequestStartTransactionRequest message (while AuthorizeRemoteStart is true), before the EV driver connects the EV and EVSE (within the connectionTimeout). The Charging Station has to authorize beforehand like a local action to start a transaction. |
| Prerequisite(s) | AuthEnabled is NOT implemented with mutability ReadOnly and the value set to false |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|---|---|
| Charging Station | CSMS |
| <u>Manual Action</u> : Trigger the CSMS to request the Charging Station to start a transaction. | |
| 2. The Test System responds with a RequestStartTransactionResponse with status <i>Accepted</i> transactionId is omitted. | 1. The CSMS sends a RequestStartTransactionRequest |
| 3. The Test System sends a AuthorizeRequest . with idToken.idToken <Configured valid_idtoken_idtoken> idToken.type <Configured valid_idtoken_type> | 4. The CSMS responds with a AuthorizeResponse . |
| 5. The Test System sends a TransactionEventRequest . with triggerReason is <i>RemoteStart</i> transactionInfo.remoteStartId is <By Test System generated remoteStartID> eventType is <i>Started</i> | 6. The CSMS responds with a TransactionEventResponse . |
| 7. Execute Reusable State <i>EnergyTransferStarted</i> (State is <i>Authorized</i> and <i>_EVConnected</i> = false) | |

| |
|--|
| Tool validations |
| <p>* Step 1: Message: RequestStartTransactionRequest - idToken.idToken <Configured valid_idtoken_idtoken> - idToken.type <Configured valid_idtoken_type></p> <p>* Step 4: Message: AuthorizeResponse - idTokenInfo.status must be <i>Accepted</i></p> |
| Post scenario validations: N/a |

TC_F_03_CSMS: Remote start transaction - Remote start first - AuthorizeRemoteStart is false

| | |
|-------------------|--|
| Test case name | Remote start transaction - Remote start first - AuthorizeRemoteStart is false |
| Test case Id | TC_F_03_CSMS |
| Use case Id(s) | F02 |
| Requirement(s) | F02.FR.01, F01.FR.02 |
| System under test | CSMS |
| Description | OCPP 2.x.x allows an EV driver to either first wait for/trigger a RequestStartTransactionRequest OR connect the EV and EVSE. Both sequences will result in being able to charge. |
| Purpose | To verify if the CSMS is able to handle a Charging Station that starts a charging session when the Charging Stations receives a RequestStartTransactionRequest message (while AuthorizeRemoteStart is false), before the EV driver connects the EV and EVSE (within the connectionTimeout). The Charging station does NOT have to authorize beforehand like a local action to start a transaction. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|--|---|
| Charging Station | CSMS |
| <u>Manual Action:</u> Trigger the CSMS to request the Charging Station to start a transaction. | |
| 2. The Test System responds with a RequestStartTransactionResponse with status <i>Accepted</i> transactionId is omitted. | 1. The CSMS sends a RequestStartTransactionRequest |
| 3. The Test System sends a TransactionEventRequest . with triggerReason is <i>RemoteStart</i> transactionInfo.remoteStartId is <i><By Test System generated remoteStartID></i> eventType is <i>Started</i> | 4. The CSMS responds with a TransactionEventResponse . |
| 5. Execute Reusable State <i>EnergyTransferStarted</i> (State is <i>Authorized</i> and <i>_EVConnected</i> = false) | |

| |
|---|
| Tool validations |
| * Step 1: Message: RequestStartTransactionRequest - idToken.idToken <i><Configured valid_idtoken_idtoken></i> - idToken.type <i><Configured valid_idtoken_type></i> |
| Post scenario validations: N/a |

TC_F_04_CSMS: Remote start transaction - Remote start first - Cable plugin timeout

| | |
|-------------------|--|
| Test case name | Remote start transaction - Remote start first - Cable plugin timeout |
| Test case Id | TC_F_04_CSMS |
| Use case Id(s) | F02, E03 |
| Requirement(s) | E03.FR.04, E03.FR.05 |
| System under test | CSMS |
| Description | OCPP 2.x.x allows an EV driver to either first wait for/trigger a RequestStartTransactionRequest OR connect the EV and EVSE. Both sequences will result in being able to charge. |
| Purpose | To verify if the CSMS is able to handle a Charging Station that deauthorizes the transaction after the EVConnectTimeout has been reached. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|--|---|
| Charging Station | CSMS |
| <u>Manual Action:</u> <i>Trigger the CSMS to request the Charging Station to start a transaction.</i> | |
| 2. The Test System responds with a RequestStartTransactionResponse with status <i>Accepted</i> transactionId is omitted. | 1. The CSMS sends a RequestStartTransactionRequest |
| 3. The Test System sends a TransactionEventRequest . with triggerReason is <i>RemoteStart</i> transactionInfo.remoteStartId is <i><By Test System generated remoteStartID></i> eventType is <i>Started</i> | 4. The CSMS responds with a TransactionEventResponse . |
| 5. The Test System sends a TransactionEventRequest . with triggerReason is <i>EVConnectTimeout</i> eventType is <i>Updated</i> | 6. The CSMS responds with a TransactionEventResponse . |
| <u>Note(s):</u> - <i>This step will be executed after the _<Configured Transaction Duration> has been reached._</i> | |

| |
|---|
| Tool validations |
| * Step 1: Message: RequestStartTransactionRequest - idToken.idToken <i><Configured valid_idtoken_idtoken></i> - idToken.type <i><Configured valid_idtoken_type></i> |
| Post scenario validations: N/a |

TC_F_06_CSMS: Remote unlock Connector - Without ongoing transaction - Accepted

| | |
|-------------------|---|
| Test case name | Remote unlock Connector - Without ongoing transaction - Accepted |
| Test case Id | TC_F_06_CSMS |
| Use case Id(s) | F05 |
| Requirement(s) | n/a |
| System under test | CSMS |
| Description | This test case describes how the CSMS can be requested to send an UnlockConnectorRequest to the charging station. It sometimes happens that a connector of a Charging Station socket does not unlock correctly. This happens most of the time when there is tension on the charging cable. This means the driver cannot unplug his charging cable from the Charging Station. To help a driver, the CSO can send a UnlockConnectorRequest to the Charging Station. The Charging Station will then try to unlock the connector again. |
| Purpose | To verify if the CSMS is able to perform the remote unlock connector mechanism as described at the OCPP specification. |
| Prerequisite(s) | |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a UnlockConnectorResponse with status <i>Unlocked</i> | 1. The CSMS sends a UnlockConnectorRequest |

| |
|--|
| Tool validations |
| * Step 1: Message UnlockConnectorRequest - evseld <Configured evseld> - connectorId <Configured connectorId> |
| Post scenario validations: - N/a |

TC_F_11_CSMS: Trigger message - MeterValues - Specific EVSE

| | |
|-------------------|--|
| Test case name | Trigger message - MeterValues - Specific EVSE |
| Test case Id | TC_F_11_CSMS |
| Use case Id(s) | F06 |
| Requirement(s) | F06.FR.01,F06.FR.02 |
| System under test | CSMS |
| Description | The CSMS can request a Charging Station to send Charging Station-initiated messages. In the request the CSMS indicates which message it wishes to receive. |
| Purpose | To verify if the CSMS is able to trigger the Charging Station to send a MeterValuesRequest for a specific EVSE, using a TriggerMessageRequest. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a TriggerMessageResponse with status <i>Accepted</i> | 1. The CSMS sends a TriggerMessageRequest |
| 3. The Test System sends a MeterValuesRequest With evseld <Configured evseld> meterValue[0].sampledValue.context <i>Trigger</i> | 4. The CSMS responds with a MeterValuesResponse |

| |
|--|
| Tool validations |
| * Step 1: Message: TriggerMessageRequest - requestedMessage must be <i>MeterValues</i> - evse.id must be <Configured evseld> |
| Post scenario validations: N/a |

TC_F_12_CSMS: Trigger message - MeterValues - All EVSE

| | |
|-------------------|--|
| Test case name | Trigger message - MeterValues - All EVSE |
| Test case Id | TC_F_12_CSMS |
| Use case Id(s) | F06 |
| Requirement(s) | F06.FR.01 |
| System under test | CSMS |
| Description | The CSMS can request a Charging Station to send Charging Station-initiated messages. In the request the CSMS indicates which message it wishes to receive. |
| Purpose | To verify if the CSMS is able to trigger the Charging Station to send a MeterValuesRequest for all EVSE, using a TriggerMessageRequest. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a TriggerMessageResponse with status <i>Accepted</i> | 1. The CSMS sends a TriggerMessageRequest |
| 3. The Test System sends a MeterValuesRequest With evseId omitted meterValue[0].sampledValue.context <i>Trigger</i> | 4. The CSMS responds with a MeterValuesResponse |
| <u>Note(s):</u> - This step will be executed for every EVSE. | |

| |
|--|
| Tool validations |
| * Step 1: Message: TriggerMessageRequest - requestedMessage must be <i>MeterValues</i> |
| Post scenario validations: N/a |

TC_F_13_CSMS: Trigger message - TransactionEvent - Specific EVSE

| | |
|-------------------|--|
| Test case name | Trigger message - TransactionEvent - Specific EVSE |
| Test case Id | TC_F_13_CSMS |
| Use case Id(s) | F06 |
| Requirement(s) | F06.FR.01,F06.FR.02 |
| System under test | CSMS |
| Description | The CSMS can request a Charging Station to send Charging Station-initiated messages. In the request the CSMS indicates which message it wishes to receive. |
| Purpose | To verify if the CSMS is able to trigger the Charging Station to send a TransactionEventRequest for a specific EVSE, using a TriggerMessageRequest. |
| Prerequisite(s) | N/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a TriggerMessageResponse with status <i>Accepted</i> | 1. The CSMS sends a TriggerMessageRequest |
| 3. The Test System sends a TransactionEventRequest With evse.id <Configured evseld> triggerReason <i>Trigger</i> transactionInfo.chargingState <i>Charging</i> meterValue is present meterValue[0].sampledValue.context <i>Trigger</i> | 4. The CSMS responds with a TransactionEventResponse |

| |
|---|
| Tool validations |
| * Step 1: Message: TriggerMessageRequest - requestedMessage must be <i>TransactionEvent</i> - evse.id must be <Configured evseld> |
| Post scenario validations: N/a |

TC_F_14_CSMS: Trigger message - TransactionEvent - All EVSE

| | |
|-------------------|--|
| Test case name | Trigger message - TransactionEvent - All EVSE |
| Test case Id | TC_F_14_CSMS |
| Use case Id(s) | F06 |
| Requirement(s) | F06.FR.01 |
| System under test | CSMS |
| Description | The CSMS can request a Charging Station to send Charging Station-initiated messages. In the request the CSMS indicates which message it wishes to receive. |
| Purpose | To verify if the CSMS is able to trigger the Charging Station to send a TransactionEventRequest for all EVSE, using a TriggerMessageRequest. |
| Prerequisite(s) | N/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a TriggerMessageResponse with status <i>Accepted</i> | 1. The CSMS sends a TriggerMessageRequest |
| 3. The Test System sends a TransactionEventRequest With evse.id omitted triggerReason <i>Trigger</i> transactionInfo.chargingState <i>Charging</i> meterValue is present meterValue[0].sampledValue.context <i>Trigger</i> | 4. The CSMS responds with a TransactionEventResponse |
| Note(s): - This step will be executed for every EVSE. | |

| |
|---|
| Tool validations |
| * Step 1: Message: TriggerMessageRequest - requestedMessage must be <i>TransactionEvent</i> |
| Post scenario validations: N/a |

TC_F_15_CSMS: Trigger message - LogStatusNotification - Idle

| | |
|-------------------|--|
| Test case name | Trigger message - LogStatusNotification - Idle |
| Test case Id | TC_F_15_CSMS |
| Use case Id(s) | F06 |
| Requirement(s) | F06.FR.01 |
| System under test | CSMS |
| Description | The CSMS can request a Charging Station to send Charging Station-initiated messages. In the request the CSMS indicates which message it wishes to receive. |
| Purpose | To verify if the CSMS is able to trigger the Charging Station to send a LogStatusNotificationRequest, using a TriggerMessageRequest. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|---|--|
| Charging Station | CSMS |
| 2. The Test System responds with a TriggerMessageResponse with status <i>Accepted</i> | 1. The CSMS sends a TriggerMessageRequest |
| 3. The Test System sends a LogStatusNotificationRequest with status <i>Idle</i> | 4. The CSMS responds with a LogStatusNotificationResponse |

| |
|--|
| Tool validations |
| * Step 1: Message: TriggerMessageRequest - requestedMessage must be <i>LogStatusNotification</i> |
| Post scenario validations: N/a |

TC_F_18_CSMS: Trigger message - FirmwareStatusNotification - Idle

| | |
|-------------------|--|
| Test case name | Trigger message - FirmwareStatusNotification - Idle |
| Test case Id | TC_F_18_CSMS |
| Use case Id(s) | F06 |
| Requirement(s) | F06.FR.01 |
| System under test | CSMS |
| Description | The CSMS can request a Charging Station to send Charging Station-initiated messages. In the request the CSMS indicates which message it wishes to receive. |
| Purpose | To verify if the CSMS is able to trigger the Charging Station to send a FirmwareStatusNotificationRequest, using a TriggerMessageRequest. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|--|---|
| Charging Station | CSMS |
| 2. The Test System responds with a TriggerMessageResponse with status <i>Accepted</i> | 1. The CSMS sends a TriggerMessageRequest |
| 3. The Test System sends a FirmwareStatusNotificationRequest with status <i>Idle</i> | 4. The CSMS responds with a FirmwareStatusNotificationResponse |

| |
|---|
| Tool validations |
| * Step 1: Message: TriggerMessageRequest - requestedMessage must be <i>FirmwareStatusNotification</i> |
| Post scenario validations: N/a |

TC_F_20_CSMS: Trigger message - Heartbeat

| | |
|-------------------|--|
| Test case name | Trigger message - Heartbeat |
| Test case Id | TC_F_20_CSMS |
| Use case Id(s) | F06 |
| Requirement(s) | F06.FR.01 |
| System under test | CSMS |
| Description | The CSMS can request a Charging Station to send Charging Station-initiated messages. In the request the CSMS indicates which message it wishes to receive. |
| Purpose | To verify if the CSMS is able to trigger the Charging Station to send a HeartbeatRequest, using a TriggerMessageRequest. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a TriggerMessageResponse with status <i>Accepted</i> | 1. The CSMS sends a TriggerMessageRequest |
| 3. The Test System sends a HeartbeatRequest | 4. The CSMS responds with a HeartbeatResponse |

| |
|--|
| Tool validations |
| * Step 1: Message: TriggerMessageRequest - requestedMessage must be <i>Heartbeat</i> |
| Post scenario validations: N/a |

TC_F_23_CSMS: Trigger message - StatusNotification - Specific EVSE - Available

| | |
|-------------------|---|
| Test case name | Trigger message - StatusNotification - Specific EVSE - Available |
| Test case Id | TC_F_23_CSMS |
| Use case Id(s) | F06 |
| Requirement(s) | F06.FR.01,F06.FR.02,F06.FR.13 |
| System under test | CSMS |
| Description | The CSMS can request a Charging Station to send Charging Station-initiated messages. In the request the CSMS indicates which message it wishes to receive. |
| Purpose | To verify if the CSMS is able to trigger the Charging Station to send a StatusNotificationRequest for a specific available EVSE, using a TriggerMessageRequest. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a TriggerMessageResponse with status <i>Accepted</i> | 1. The CSMS sends a TriggerMessageRequest |
| 3. The Test System notifies the CSMS about the current state of the connector. Message: StatusNotificationRequest - connectorStatus <i>Available</i> - evseld <i><Configured evseld></i> - connectorId <i><Configured connectorId></i> Message: NotifyEventRequest - trigger <i>Delta</i> - actualValue <i>"Available"</i> - component.name <i>"Connector"</i> - component.evse.id <i><Configured evseld></i> - component.evse.connectorId <i><Configured connectorId></i> - variable.name <i>"AvailabilityState"</i> | 4. The CSMS responds accordingly. |

| |
|--|
| Tool validations |
| * Step 1: Message: TriggerMessageRequest - requestedMessage must be <i>StatusNotification</i> - evse.id must be <i><Configured evseld></i> |
| Post scenario validations: N/a |

TC_F_24_CSMS: Trigger message - StatusNotification - Specific EVSE - Occupied

| | |
|-------------------|--|
| Test case name | Trigger message - StatusNotification - Specific EVSE - Occupied |
| Test case Id | TC_F_24_CSMS |
| Use case Id(s) | F06 |
| Requirement(s) | F06.FR.01,F06.FR.02,F06.FR.13 |
| System under test | CSMS |
| Description | The CSMS can request a Charging Station to send Charging Station-initiated messages. In the request the CSMS indicates which message it wishes to receive. |
| Purpose | To verify if the CSMS is able to trigger the Charging Station to send a StatusNotificationRequest for a specific occupied EVSE, using a TriggerMessageRequest. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <p>1. The Test System notifies the CSMS about the current state of the connector.</p> <p>Message: StatusNotificationRequest</p> <ul style="list-style-type: none"> - connectorStatus <i>Occupied</i> - evseld <Configured evseld> - connectorId <Configured connectorId> <p>Message: NotifyEventRequest</p> <ul style="list-style-type: none"> - trigger <i>Delta</i> - actualValue <i>"Occupied"</i> - component.name <i>"Connector"</i> - component.evse.id <Configured evseld> - component.evse.connectorId <Configured connectorId> - variable.name <i>"AvailabilityState"</i> | <p>2. The CSMS responds accordingly.</p> |
| <p>4. The Test System responds with a TriggerMessageResponse with status <i>Accepted</i></p> | <p>3. The CSMS sends a TriggerMessageRequest</p> |
| <p>5. The Test System notifies the CSMS about the current state of the connector.</p> <p>Message: StatusNotificationRequest</p> <ul style="list-style-type: none"> - connectorStatus <i>Occupied</i> - evseld <Configured evseld> - connectorId <Configured connectorId> <p>Message: NotifyEventRequest</p> <ul style="list-style-type: none"> - trigger <i>Delta</i> - actualValue <i>"Occupied"</i> - component.name <i>"Connector"</i> - component.evse.id <Configured evseld> - component.evse.connectorId <Configured connectorId> - variable.name <i>"AvailabilityState"</i> | <p>6. The CSMS responds accordingly.</p> |

| Tool validations |
|---|
| <p>* Step 1:</p> <p>Message: TriggerMessageRequest</p> <ul style="list-style-type: none">- requestedMessage must be <i>StatusNotification</i>- evse.id must be <i><Configured evseld></i> |
| <p>Post scenario validations:</p> <p>N/a</p> |

TC_F_27_CSMS: Trigger message - NotImplemented

| | |
|--------------------------|--|
| Test case name | Trigger message - NotImplemented |
| Test case Id | TC_F_27_CSMS |
| Use case Id(s) | F06 |
| Requirement(s) | F06.FR.08 |
| System under test | CSMS |
| Description | The CSMS can request a Charging Station to send Charging Station-initiated messages. In the request the CSMS indicates which message it wishes to receive. |
| Purpose | To verify if the CSMS is able to handle a Charging Station that does not support the requested message value from a TriggerMessageRequest. |
| Prerequisite(s) | N/a |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a TriggerMessageResponse with status <i>NotImplemented</i> | 1. The CSMS sends a TriggerMessageRequest |

| |
|--|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

G Availability**TC_G_03_CSMS: Change Availability EVSE - Operative to inoperative**

| | |
|--------------------------|--|
| Test case name | Change Availability EVSE - Operative to inoperative |
| Test case Id | TC_G_03_CSMS |
| Use case Id(s) | G03 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | This test case covers how the CSMS requests the Charging Station to change the availability of one of the EVSEs from Operative to Inoperative. An EVSE is considered Operative in any status other than Faulted and Unavailable. |
| Purpose | To verify if the CSMS is able to perform the change availability mechanism as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>Unavailable</i> for <Configured evseld> | |

| |
|--|
| Tool validations |
| N/a |
| Post scenario validations: - N/a |

TC_G_04_CSMS: Change Availability EVSE - Inoperative to operative

| | |
|-------------------|--|
| Test case name | Change Availability EVSE - Inoperative to operative |
| Test case Id | TC_G_04_CSMS |
| Use case Id(s) | G03 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | This test case covers how the CSMS requests the Charging Station to change the availability of one of the EVSEs from Inoperative to Operative. An EVSE is considered Operative in any status other than Faulted and Unavailable. |
| Purpose | To verify if the CSMS is able to perform the change availability mechanism as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: <i>Unavailable</i> for <Configured evseld> |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|--|--|
| Charging Station | CSMS |
| <u>Manual Action</u> : Request the CSMS to change the availability of an EVSE to Operative. | |
| 2. The Test System responds with a ChangeAvailabilityResponse with status <i>Accepted</i> | 1. The CSMS sends a ChangeAvailabilityRequest |
| 3. The Test System notifies the CSMS about the current state of all connectors belonging to the specified EVSE (and optionally also from the EVSE itself). Message: StatusNotificationRequest - connectorStatus <i>Available</i> - evseld <Configured evseld> Message: NotifyEventRequest - trigger <i>Delta</i> - actualValue "Available" - component.name "EVSE" / Connector - component.evse.id <Configured evseld> - variable.name "AvailabilityState" | 4. The CSMS responds accordingly. |

| |
|--|
| Tool validations |
| * Step 1: Message ChangeAvailabilityRequest - operationalStatus <i>Operative</i> - evse.id <Configured evseld> - connectorId <i>omit</i> |
| Post scenario validations: - N/a |

TC_G_05_CSMS: Change Availability Charging Station - Operative to inoperative

| | |
|-------------------|--|
| Test case name | Change Availability Charging Station - Operative to inoperative |
| Test case Id | TC_G_05_CSMS |
| Use case Id(s) | G04 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | <p>This test case describes how the CSMS requests the Charging Station to change the availability from operative to inoperative.</p> <p>A Charging Station is considered Operative when it is charging or ready for charging.</p> <p>A Charging Station is considered Inoperative when it does not allow any charging.</p> |
| Purpose | To verify if the CSMS is able to perform the change availability mechanism as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|---|--|
| Charging Station | CSMS |
| <u>Manual Action:</u> Request the CSMS to change the availability of the Charging Station to Inoperative. | |
| 2. The Test System responds with a ChangeAvailabilityResponse with status <i>Accepted</i> | 1. The CSMS sends a ChangeAvailabilityRequest |
| 3. The Test System notifies the CSMS about the current state of all connectors Message: StatusNotificationRequest - connectorStatus <i>Unavailable</i> Message: NotifyEventRequest - trigger <i>Delta</i> - actualValue <i>"Unavailable"</i> - component.name <i>"Connector"</i> - variable.name <i>"AvailabilityState"</i> | 4. The CSMS responds accordingly. |

| |
|---|
| Tool validations |
| * Step 1: Message ChangeAvailabilityRequest - operationalStatus <i>Inoperative</i> - evseld <i>omit</i> - connectorId <i>omit</i> |
| Post scenario validations: - N/a |

TC_G_06_CSMS: Change Availability Charging Station - Inoperative to operative

| | |
|-------------------|--|
| Test case name | Change Availability Charging Station - Inoperative to operative |
| Test case Id | TC_G_06_CSMS |
| Use case Id(s) | G04 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | <p>This test case describes how the CSMS requests the Charging Station to change the availability from inoperative to operative.</p> <p>A Charging Station is considered Operative when it is charging or ready for charging.</p> <p>A Charging Station is considered Inoperative when it does not allow any charging.</p> |
| Purpose | To verify if the CSMS is able to perform the change availability mechanism as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): Charging Station set to <i>Unavailable</i> (Original status was Available) |

| Main (Test scenario) | |
|--|--|
| Charging Station | CSMS |
| <u>Manual Action</u> : Request the CSMS to change the availability of the Charging Station to Inoperative. | |
| 2. The Test System responds with a ChangeAvailabilityResponse with status <i>Accepted</i> | 1. The CSMS sends a ChangeAvailabilityRequest |
| 3. The Test System notifies the CSMS about the current state of all connectors. Message: StatusNotificationRequest - connectorStatus <i>Available</i> Message: NotifyEventRequest - trigger <i>Delta</i> - actualValue <i>"Available"</i> - component.name <i>"Connector"</i> - variable.name <i>"AvailabilityState"</i> | 4. The CSMS responds accordingly. |

| |
|---|
| Tool validations |
| * Step 1: Message ChangeAvailabilityRequest - operationalStatus <i>Operative</i> - evseld <i>omit</i> - connectorId <i>omit</i> |
| Post scenario validations: - N/a |

TC_G_07_CSMS: Change Availability Connector - Operative to inoperative

| | |
|-------------------|---|
| Test case name | Change Availability Connector - Operative to inoperative |
| Test case Id | TC_G_07_CSMS |
| Use case Id(s) | G03 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | This test case covers how the CSMS requests the Charging Station to change the availability of one of the Connectors of one EVSE from Operative to Inoperative. A Connector is considered Operative in any status other than Faulted and Unavailable. |
| Purpose | To verify if the CSMS is able to perform the change availability mechanism as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action:</u> Request the CSMS to change the availability of a Connector to Inoperative. | |
| 2. The Test System responds with a ChangeAvailabilityResponse with status <i>Accepted</i> | 1. The CSMS sends a ChangeAvailabilityRequest |
| 3. The Test System notifies the CSMS about the current state of the connector. Message: StatusNotificationRequest - connectorStatus <i>Unavailable</i> - evseld <i><Configured evseld></i> - connectorId <i><Configured connectorId></i> Message: NotifyEventRequest - trigger <i>Delta</i> - actualValue <i>"Unavailable"</i> - component.name <i>"Connector"</i> - component.evse.id <i><Configured evseld></i> - component.evse.connectorid <i><Configured connectorId></i> - variable.name <i>"AvailabilityState"</i> | 4. The CSMS responds accordingly. |

| |
|--|
| Tool validations |
| * Step 1: Message ChangeAvailabilityRequest - operationalStatus <i>Inoperative</i> - evse.id <i><Configured evseld></i> - evse.connectorId <i><Configured connectorId></i> |
| Post scenario validations: N/a |

TC_G_08_CSMS: Change Availability Connector - Inoperative to operative

| | |
|-------------------|---|
| Test case name | Change Availability Connector - Inoperative to operative |
| Test case Id | TC_G_08_CSMS |
| Use case Id(s) | G03 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | This test case covers how the CSMS requests the Charging Station to change the availability of one of the Connectors from one EVSE from Inoperative to Operative. A Connector is considered Operative in any status other than Faulted and Unavailable. |
| Purpose | To verify if the CSMS is able to perform the change availability mechanism as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: <i>Unavailable</i> for <Configured connectorId> |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|--|--|
| Charging Station | CSMS |
| <u>Manual Action</u> : Request the CSMS to change the availability of a Connector to Operative. | |
| 2. The Test System responds with a ChangeAvailabilityResponse with status Accepted | 1. The CSMS sends a ChangeAvailabilityRequest |
| 3. The Test System notifies the CSMS about the current state of the connector. Message: StatusNotificationRequest - connectorStatus Available - evseld <Configured evseld> - connectorId <Configured connectorId> Message: NotifyEventRequest - trigger Delta - actualValue "Available" - component.name "Connector" - component.evse.id <Configured evseld> - component.evse.connectorId <Configured connectorId> - variable.name "AvailabilityState" | 4. The CSMS responds accordingly. |

| |
|--|
| Tool validations |
| * Step 1: Message ChangeAvailabilityRequest - operationalStatus <i>Operative</i> - evse.id <Configured evseld> - evse.connectorId <Configured connectorId> |
| Post scenario validations: N/a |

TC_G_11_CSMS: Change Availability EVSE - With ongoing transaction

| | |
|-------------------|--|
| Test case name | Change Availability EVSE - With ongoing transaction |
| Test case Id | TC_G_11_CSMS |
| Use case Id(s) | G03 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | This test case covers how the CSMS requests the Charging Station to change the availability of one of the EVSEs from Operative to Inoperative. An EVSE is considered Operative in any status other than Faulted and Unavailable. |
| Purpose | To verify if the CSMS is able to send a change availability request during a transaction according to the mechanism as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: State is <i>EnergyTransferStarted</i> |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Note(s)</u> : Request the CSMS to change the availability to inoperative | |
| 2. The Test System responds with a ChangeAvailabilityResponse with status <i>Scheduled</i> | 1. The CSMS sends a ChangeAvailabilityRequest |
| <u>Note(s)</u> : Wait for <Configured Transaction Duration> | |
| 3. Execute Reusable State <i>StopAuthorized</i> | |
| 4. Execute Reusable State <i>EVConnectedPostSession</i> | |
| 5. Execute Reusable State <i>EVDisconnected</i> | |
| 6. The Test System notifies the CSMS about the current state of all connectors with Message: StatusNotificationRequest - connectorStatus <i>Unavailable</i> - evseld <Configured evseld> OR Message: NotifyEventRequest - trigger <i>Delta</i> - actualValue "Unavailable" - component.name "Connector" - component.evse.id <Configured evseld> - variable.name "AvailabilityState" | 7. The CSMS responds accordingly. |

| |
|---|
| Tool validations |
| * Step 1: Message ChangeAvailabilityRequest - operationalStatus <i>Inoperative</i> - evse.id <Configured evseld> - connectorId omit |
| Post scenario validations: - A respond to report the state of a connector has been received for all connectors. |

TC_G_14_CSMS: Change Availability Charging Station - With ongoing transaction

| | |
|-------------------|--|
| Test case name | Change Availability Charging Station - With ongoing transaction |
| Test case Id | TC_G_14_CSMS |
| Use case Id(s) | G04 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | This test case covers how the CSMS requests the Charging Station to change the availability of one of the EVSEs from Operative to Inoperative. An EVSE is considered Operative in any status other than Faulted and Unavailable. |
| Purpose | To verify if the CSMS is able to send a change availability request during a transaction according to the mechanism as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: State is <i>EnergyTransferStarted</i> |

| Main (Test scenario) | |
|---|--|
| Charging Station | CSMS |
| <u>Note(s)</u> : Request the CSMS to change the availability of the station to inoperative | |
| 2. The Test System responds with a ChangeAvailabilityResponse with status <i>Scheduled</i> | 1. The CSMS sends a ChangeAvailabilityRequest |
| 3. The Test System notifies the CSMS about the current state of all unoccupied connectors with Message: StatusNotificationRequest - connectorStatus <i>Unavailable</i> | 4. The CSMS responds accordingly. |
| <u>Note(s)</u> : Wait for <Configured Transaction Duration> | |
| 5. Execute Reusable State <i>StopAuthorized</i> | |
| 6. Execute Reusable State <i>EVConnectedPostSession</i> | |
| 7. Execute Reusable State <i>EVDisconnected</i> | |
| 8. The Test System notifies the CSMS about the current state of the configured connector with Message: StatusNotificationRequest - connectorStatus <i>Unavailable</i> | 9. The CSMS responds accordingly. |

| |
|---|
| Tool validations |
| * Step 1: Message ChangeAvailabilityRequest - operationalStatus <i>Inoperative</i> - evseld <i>omit</i> - connectorId <i>omit</i> |
| Post scenario validations: - A respond to report the state of a connector has been received for all connectors. |

TC_G_17_CSMS: Change Availability Connector - With ongoing transaction

| | |
|-------------------|--|
| Test case name | Change Availability Connector - With ongoing transaction |
| Test case Id | TC_G_17_CSMS |
| Use case Id(s) | G03 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | This test case covers how the CSMS requests the Charging Station to change the availability of one of the EVSEs from Operative to Inoperative. An EVSE is considered Operative in any status other than Faulted and Unavailable. |
| Purpose | To verify if the CSMS is able to send a change availability request during a transaction according to the mechanism as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: State is <i>EnergyTransferStarted</i> |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Note(s)</u> : Request the CSMS to change the availability of one connector to inoperative | |
| 2. The Test System responds with a ChangeAvailabilityResponse with status <i>Scheduled</i> | 1. The CSMS sends a ChangeAvailabilityRequest |
| <u>Note(s)</u> : Wait for <Configured Transaction Duration> | |
| 3. Execute Reusable State <i>StopAuthorized</i> | |
| 4. Execute Reusable State <i>EVConnectedPostSession</i> | |
| 5. Execute Reusable State <i>EVDisconnected</i> | |
| 6. The Test System notifies the CSMS about the current state of all connectors with Message: StatusNotificationRequest - connectorStatus <i>Unavailable</i> - evseId <Configured evseId> - connectorId <Configured connectorId> | 7. The CSMS responds accordingly. |

| |
|--|
| Tool validations |
| * Step 1: Message ChangeAvailabilityRequest - operationalStatus <i>Inoperative</i> - evse.id <Configured evseId> - evse.connectorId <Configured connectorId> |
| Post scenario validations: - A respond to report the state of a connector has been received for all connectors. |

TC_G_20_CSMS: Connector status Notification - Lock Failure

| | |
|-------------------|--|
| Test case name | Connector status Notification - Lock Failure |
| Test case Id | TC_G_20_CSMS |
| Use case Id(s) | G05 |
| Requirement(s) | G05.FR.03 |
| System under test | CSMS |
| Description | This test case describes how the EV Driver is prevented from starting a charge session at the Charging Station while the Connector is not locked properly. |
| Purpose | To verify if the CSMS responds on a notifyeventrequest as described at the OCPP specification. |
| Prerequisite(s) | - N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Test System sends a NotifyEventRequest with - eventData.trigger <i>Delta</i> - eventData.component.name <i>"ConnectorPlugRetentionLock"</i> - eventData.variable.name <i>"Problem"</i> - eventData.actualValue <i>"true"</i> | 2. The CSMS responds with a NotifyEventResponse |

| |
|-------------------------------------|
| Tool validations |
| N/a |
| Post scenario validations: - N/a |

H Reservation

TC_H_01_CSMS: Reserve a specific EVSE - Accepted - Valid idToken

| | |
|-------------------|---|
| Test case name | Reserve a specific EVSE - Accepted - Valid idToken |
| Test case Id | TC_H_01_CSMS |
| Use case Id(s) | H01(S2), H03 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | The CSMS is able to reserve a specific EVSE for a specific IdToken by sending a ReserveNowRequest containing an evseld. |
| Purpose | To verify if the CSMS is able to request the Charging Station to reserve a specific EVSE, until the EV Driver with the specified IdToken arrives. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>Reserved</i> for <Configured evseld> | |

| |
|-----------------------------------|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_H_07_CSMS: Reserve a specific EVSE - Reservation Ended / not used

| | |
|-------------------|--|
| Test case name | Reserve a specific EVSE - Reservation Ended / not used |
| Test case Id | TC_H_07_CSMS |
| Use case Id(s) | H01(S2), H04 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | The CSMS is able to reserve a specific EVSE for a specific IdToken by sending a ReserveNowRequest containing an evseld. |
| Purpose | To verify if the CSMS is able to handle a reservation that is canceled by the Charging Station, because the EV driver did not arrive before the set expiryDateTime was reached. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action:</u> Trigger the CSMS to send a ReserveNowRequest for a specific EVSE. | |
| 2. The Test System responds with a ReserveNowResponse With status <i>Accepted</i> | 1. The CSMS sends a ReserveNowRequest with expiryDateTime <i>current time + <configured transaction duration></i> |
| 3. The Test System notifies the CSMS about the current state of the connector(s) of the configured EVSE Message: StatusNotificationRequest with connectorStatus <i>Reserved</i> Message: NotifyEventRequest with trigger <i>Delta</i> actualValue <i>"Reserved"</i> component.name <i>"Connector"</i> variable.name <i>"AvailabilityState"</i> | 4. The CSMS responds accordingly. |
| 5. The Test System notifies the CSMS about the current state of the connector(s) of the configured EVSE Message: StatusNotificationRequest with connectorStatus <i>Available</i> Message: NotifyEventRequest with trigger <i>Delta</i> actualValue <i>"Available"</i> component.name <i>"Connector"</i> variable.name <i>"AvailabilityState"</i> <u>Note(s):</u> - The Test System waits until the provided expiryDateTime from step 1 expires before executing this step. | 6. The CSMS responds accordingly. |
| 7. The Test System sends a ReservationStatusUpdateRequest With reservationUpdateStatus <i>Expired</i> reservationId <i><id received at step 1></i> | 8. The CSMS responds with a ReservationStatusUpdateResponse |

| Tool validations |
|---|
| <div>* Step 1:</div> <div>Message: ReserveNowRequest</div> <div><div>- evseld must be <i><Configured evseld></i></div><div>- connectorType must be omitted</div><div>- idToken.idToken <i><Configured valid_idtoken_idtoken></i></div><div>- idToken.type <i><Configured valid_idtoken_type></i></div></div> |
| <div>Post scenario validations:</div> <div>N/a</div> |

TC_H_08_CSMS: Reserve an unspecified EVSE - Accepted

| | |
|-------------------|---|
| Test case name | Reserve an unspecified EVSE - Accepted |
| Test case Id | TC_H_08_CSMS |
| Use case Id(s) | H01(S1), H03 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | The CSMS is able to reserve an unspecified EVSE for a specific IdToken by sending a ReserveNowRequest without an evseld. |
| Purpose | To verify if the CSMS is able to request the Charging Station to reserve an unspecified EVSE, until the EV Driver with the specified IdToken arrives. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action:</u> Trigger the CSMS to send a ReserveNowRequest for an unspecified EVSE. | |
| 2. The Test System responds with a ReserveNowResponse with status Accepted | 1. The CSMS sends a ReserveNowRequest |
| 3. The Test System notifies the CSMS about the current state of the connector(s) of the configured EVSE Message: StatusNotificationRequest with connectorStatus Reserved Message: NotifyEventRequest with trigger Delta actualValue "Reserved" component.name "Connector" variable.name "AvailabilityState" <u>Note(s):</u> - The Test System will execute this step, if it is configured with only one EVSE. | 4. The CSMS responds accordingly. |

| |
|---|
| Tool validations |
| * Step 1: Message: ReserveNowRequest - evseld must be omitted - connectorType must be omitted - idToken.idToken <Configured valid_idtoken_idtoken> - idToken.type <Configured valid_idtoken_type> |
| Post scenario validations: N/a |

TC_H_14_CSMS: Reserve an unspecified EVSE - Amount of EVSEs available equals the amount of reservations

| | |
|-------------------|--|
| Test case name | Reserve an unspecified EVSE - Amount of EVSEs available equals the amount of reservations |
| Test case Id | TC_H_14_CSMS |
| Use case Id(s) | H01(S1) |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | The CSMS is able to reserve an unspecified EVSE for a specific IdToken by sending a ReserveNowRequest without an evseld. |
| Purpose | To verify if the CSMS is able to handle that the Charging Station sets all available EVSE to reserved, when the amount of EVSEs available equals the amount of reservations. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|--|--|
| Charging Station | CSMS |
| <u>Manual Action:</u> Trigger the CSMS to send a ReserveNowRequest for an unspecified EVSE. | |
| 2. The Test System responds with a ReserveNowResponse with status <i>Accepted</i> | 1. The CSMS sends a ReserveNowRequest <u>Note(s):</u> - <i>This step needs to executed time the amount of EVSE configured for the Test System.</i> |
| 3. The Test System notifies the CSMS about the current state of all connectors Message: StatusNotificationRequest with connectorStatus <i>Reserved</i> Message: NotifyEventRequest with trigger <i>Delta</i> actualValue <i>"Reserved"</i> component.name <i>"Connector"</i> variable.name <i>"AvailabilityState"</i> <u>Note(s):</u> - <i>This step will be executed after the last ReserveNowRequest has been sent from step 1.</i> | 4. The CSMS responds accordingly. |

| |
|---|
| Tool validations |
| * Step 1: Message: ReserveNowRequest - evseld must be omitted - connectorType must be omitted - idToken.idToken <Configured valid_idtoken_idtoken> - idToken.type <Configured valid_idtoken_type> |

| |
|--|
| Tool validations |
| Post scenario validations: N/a |

TC_H_15_CSMS: Reserve a connector with a specific type - Success

| | |
|-------------------|---|
| Test case name | Reserve a connector with a specific type - Success |
| Test case Id | TC_H_15_CSMS |
| Use case Id(s) | H01(S3), H03 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | The CSMS is able to reserve an EVSE with a connector with a specific type for a specific IdToken by sending a ReserveNowRequest with a connectorType. |
| Purpose | To verify if the CSMS is able to request the Charging Station to reserve an EVSE with a connector with a specific type, until the EV Driver with the specified IdToken arrives. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|--|--|
| Charging Station | CSMS |
| <u>Manual Action</u> : Trigger the CSMS to send a ReserveNowRequest for a specific ConnectorType. | |
| 2. The Test System responds with a ReserveNowResponse With status <i>Accepted</i> | 1. The CSMS sends a ReserveNowRequest |
| 3. The Test System notifies the CSMS about the current state of the connector Message: StatusNotificationRequest with connectorStatus <i>Reserved</i> Message: NotifyEventRequest with trigger <i>Delta</i> actualValue <i>"Reserved"</i> component.name <i>"Connector"</i> variable.name <i>"AvailabilityState"</i> | 4. The CSMS responds accordingly. |

| |
|---|
| Tool validations |
| * Step 1: Message: ReserveNowRequest - evseld must be omitted - connectorType must be <i><Configured connectorType></i> - idToken.idToken <i><Configured valid_idtoken_idtoken></i> - idToken.type <i><Configured valid_idtoken_type></i> |
| Post scenario validations: N/a |

TC_H_17_CSMS: Cancel reservation of an EVSE - Success

| | |
|-------------------|---|
| Test case name | Cancel reservation of an EVSE - Success |
| Test case Id | TC_H_17_CSMS |
| Use case Id(s) | H02 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | The CSMS is able to cancel a reservation by sending a CancelReservationRequest to the Charging Station. |
| Purpose | To verify if the CSMS is able to request the Charging Station to cancel a reservation, by sending a CancelReservationRequest |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action:</u> Trigger the CSMS to send a <i>ReserveNowRequest</i> for a specific EVSE. | |
| 2. The Test System responds with a ReserveNowResponse with status <i>Accepted</i> | 1. The CSMS sends a ReserveNowRequest |
| 3. The Test System notifies the CSMS about the current state of the connector(s) of the configured EVSE Message: StatusNotificationRequest with connectorStatus <i>Reserved</i> Message: NotifyEventRequest with trigger <i>Delta</i> actualValue <i>"Reserved"</i> component.name <i>"Connector"</i> variable.name <i>"AvailabilityState"</i> | 4. The CSMS responds accordingly. |
| <u>Manual Action:</u> Trigger the CSMS to send a <i>CancelReservationRequest</i> for the reservation created at step 1. | |
| 6. The Test System responds with a CancelReservationResponse With status <i>Accepted</i> | 5. The CSMS sends a CancelReservationRequest |
| 7. The Test System notifies the CSMS about the current state of the connector(s) of the configured EVSE Message: StatusNotificationRequest with connectorStatus <i>Available</i> Message: NotifyEventRequest with trigger <i>Delta</i> actualValue <i>"Available"</i> component.name <i>"Connector"</i> variable.name <i>"AvailabilityState"</i> | 8. The CSMS responds accordingly. |

| Tool validations |
|---|
| <div>* Step 1: Message: ReserveNowRequest - evseld must be <Configured evseld> - connectorType must be omitted - idToken.idToken <Configured valid_idtoken_idtoken> - idToken.type <Configured valid_idtoken_type> * Step 5: Message: CancelReservationRequest - reservationId must be equal to the id provided at step 1</div> |
| <div>Post scenario validations: N/a</div> |

TC_H_19_CSMS: Reserve a specific EVSE - Use a reserved EVSE with GroupId

| | |
|-------------------|---|
| Test case name | Reserve a specific EVSE - Use a reserved EVSE with GroupId |
| Test case Id | TC_H_19_CSMS |
| Use case Id(s) | H01, H03 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | The CSMS is able to reserve an EVSE for a specific group by sending a ReserveNowRequest containing a groupIdToken . |
| Purpose | To verify if the CSMS is able to request the Charging Station create a reservation for a specific group, by sending a ReserveNowRequest with a groupIdToken |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|--|--|
| Charging Station | CSMS |
| <u>Manual Action:</u> Trigger the CSMS to send a ReserveNowRequest with a groupIdToken for a specific EVSE. | |
| 2. The Test System responds with a ReserveNowResponse With status <i>Accepted</i> | 1. The CSMS sends a ReserveNowRequest |
| 3. The Test System notifies the CSMS about the current state of the connector(s) of the configured EVSE Message: StatusNotificationRequest with connectorStatus <i>Reserved</i> Message: NotifyEventRequest with trigger <i>Delta</i> actualValue <i>"Reserved"</i> component.name <i>"Connector"</i> variable.name <i>"AvailabilityState"</i> | 4. The CSMS responds accordingly. |

| |
|--|
| Tool validations |
| * Step 1: Message: ReserveNowRequest - evseld must be <i><Configured evseld></i> - connectorType must be omitted - groupIdToken must be provided - idToken.idToken <i><Configured valid_idtoken_idtoken></i> - idToken.type <i><Configured valid_idtoken_type></i> |
| Post scenario validations: N/a |

TC_H_20_CSMS: Charging Station cancels reservation when Faulted

| | |
|-------------------|--|
| Test case name | Charging Station cancels reservation when Faulted |
| Test case Id | TC_H_20_CSMS |
| Use case Id(s) | H01 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | The Charging Station will cancel reservations, when the EVSE specified for a reservation is set to an inoperative state. |
| Purpose | To verify if the CSMS is able to handle it when the reservation is canceled when the availability state of the EVSE specified for the reservation is set to Faulted by the Test System. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| Manual Action: Trigger the CSMS to send a ReserveNowRequest for a specific EVSE. | |
| 2. The Test System responds with a ReserveNowResponse With status <i>Accepted</i> | 1. The CSMS sends a ReserveNowRequest |
| 3. The Test System notifies the CSMS about the current state of the connector(s) of the configured EVSE Message: StatusNotificationRequest with connectorStatus <i>Reserved</i> Message: NotifyEventRequest with trigger <i>Delta</i> actualValue "Reserved" component.name "Connector" variable.name "AvailabilityState" | 4. The CSMS responds accordingly. |
| 5. The Test System notifies the CSMS about the current state of the connector(s) of the configured EVSE Message: StatusNotificationRequest with connectorStatus <i>Faulted</i> Message: NotifyEventRequest with trigger <i>Delta</i> actualValue "Faulted" component.name "Connector" variable.name "AvailabilityState" | 6. The CSMS responds accordingly. |
| 7. The Test System sends a ReservationStatusUpdateRequest With reservationUpdateStatus <i>Removed</i> reservationId <id received at step 1> | 8. The CSMS responds with a ReservationStatusUpdateResponse |

| Tool validations |
|---|
| <div>* Step 1:</div> <div>Message: ReserveNowRequest</div> <div><div>- evseld must be <i><Configured evseld></i></div><div>- connectorType must be omitted</div><div>- idToken.idToken <i><Configured valid_idtoken_idtoken></i></div><div>- idToken.type <i><Configured valid_idtoken_type></i></div></div> |
| <div>Post scenario validations:</div> <div>N/a</div> |

TC_H_22_CSMS: Reserve a specific EVSE - Configured to Reject

| | |
|-------------------|---|
| Test case name | Reserve a specific EVSE - Configured to Reject |
| Test case Id | TC_H_22_CSMS |
| Use case Id(s) | H01 |
| Requirement(s) | |
| System under test | CSMS |
| Description | The CSMS is able to reserve a specific EVSE for a specific IdToken by sending a ReserveNowRequest containing an evseld. |
| Purpose | To verify if the CSMS is able to correctly read the respond from a charging station when it is configured not to accept reservations. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a ReserveNowResponse with - status <i>Rejected</i> | 1. The CSMS sends a ReserveNowRequest |

| |
|-----------------------------------|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

I Tariff and Cost

TC_I_01_CSMS: Show EV Driver running total cost during charging - costUpdatedRequest

| | |
|-------------------|--|
| Test case name | Show EV Driver running total cost during charging - costUpdatedRequest |
| Test case Id | TC_I_01_CSMS |
| Use case Id(s) | I02 |
| Requirement(s) | I02.FR.01 |
| System under test | CSMS |
| Description | While a transaction is ongoing, the driver wants to know how much the running total cost is, updated at a relevant interval. |
| Purpose | To verify if the CSMS is able to correctly send the running total cost as described in the OCPP specification. |
| Prerequisite(s) | N/a |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State(s):

N/a

Main (Test scenario)

| | |
|---|--|
| Charging Station | CSMS |
| 1. The Test System sends an AuthorizeRequest with idToken.idToken <Configured valid_idtoken_idtoken> idToken.type <Configured valid_idtoken_type> | 2. The CSMS responds with an AuthorizeResponse |
| 3. The Test System sends a TransactionEventRequest with - triggerReason Authorized - idToken.idToken <Configured valid_idtoken_idtoken> - idToken.type <Configured valid_idtoken_type> - eventType Updated | 4. The CSMS responds with a TransactionEventResponse |
| 5. Execute Reusable State <i>EVConnectedPreSession</i> | |
| 6. Execute Reusable State <i>EnergyTransferStarted</i> | |
| 7. The Test System sends a TransactionEventRequest With triggerReason is <i>MeterValuePeriodic</i> eventType is <i>Updated</i> timestamp <The intervals between the timestamps of the received Meter Value messages equals the configured sampled Meter Values interval>. sampledValue.context is <i>Sample.Periodic</i> <u>Note(s):</u> - This step will be executed every <Configured sampled Meter Values interval> - The Test System will end the testcase after two MeterValues. | 8. The CSMS responds with a TransactionEventResponse |
| 10. The Test System responds with a CostUpdatedResponse | 9. The CSMS sends a CostUpdatedRequest <u>Note(s):</u> - This step will be executed after every <i>TransactionEventResponse</i> , if the message did not contain a <i>totalCost</i> . |

| Tool validations |
|---|
| <div>* Step 2: Message AuthorizeResponse - idTokenInfo.status <i>Accepted</i></div> <div>* Step 4: Message TransactionEventResponse - idTokenInfo.status <i>Accepted</i> - totalCost <i><Optional></i></div> <div>* Step 7: Message (Optional) CostUpdatedRequest - transactionId <i><Generated TransactionId></i></div> |
| <div>Post scenario validations: - N/a</div> |

TC_I_02_CSMS: Show EV Driver Final Total Cost After Charging

| | |
|-------------------|--|
| Test case name | Show EV Driver Final Total Cost After Charging |
| Test case Id | TC_I_02_CSMS |
| Use case Id(s) | I03 |
| Requirement(s) | I03.FR.02 |
| System under test | CSMS |
| Description | While a transaction is ongoing, the driver wants to know how much the running total cost is, updated at a relevant interval. |
| Purpose | To verify if the CSMS is able to correctly send the total cost as described in the OCPP specification. |
| Prerequisite(s) | - N/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: - CSMS is configured with a tariff which is based on energy consumed. |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|--|---|
| Charging Station | CSMS |
| 1. Execute Reusable State <i>EVConnectedPreSession</i> | |
| <ul style="list-style-type: none">- The TransactionEventRequest contains the MeterValue field.- sampledValue[0].value 1000- sampledValue[0].context <i>Transaction.Begin</i> | |
| 2. Execute Reusable State <i>Authorized</i> | |
| 3. Execute Reusable State <i>EnergyTransferStarted</i> | |
| 4. Execute Reusable State <i>StopAuthorized</i> | |
| 5. Execute Reusable State <i>EVConnectedPostSession</i> | |
| 6. The Test System notifies the CSMS about the current state of the configured connector. Message: StatusNotificationRequest <ul style="list-style-type: none">- connectorStatus <i>Available</i> Message: NotifyEventRequest <ul style="list-style-type: none">- trigger <i>Delta</i>- actualValue <i>"Available"</i>- component.name <i>"Connector"</i>- variable.name <i>"AvailabilityState"</i> | 7. The CSMS responds accordingly. |
| 8. The Test System sends a TransactionEventRequest with <ul style="list-style-type: none">- triggerReason <i>EVCommunicationLost</i>- eventType <i>Ended</i>- transactionInfo.chargingState <i>Idle</i>- transactionInfo.stoppedReason <i>EVDisconnected</i>- meterValue[0].sampledValue[0].value 6000- meterValue[0].sampledValue[0].context <i>Transaction.End</i> | 9. The CSMS responds with a TransactionEventResponse |

| |
|--|
| Tool validations |
| * Step 9: Message TransactionEventResponse - totalCost <Not omitted> |
| Post scenario validations: - N/a |

J Meter Values

TC_J_01_CSMS: Clock-aligned Meter Values - No transaction ongoing

| | |
|-------------------|--|
| Test case name | Clock-aligned Meter Values - No transaction ongoing |
| Test case Id | TC_J_01_CSMS |
| Use case Id(s) | J01 |
| Requirement(s) | J01.FR.18 |
| System under test | CSMS |
| Description | The Charging Station samples the electrical meter or other sensor/transducer hardware to provide information about its Meter Values. Depending on configuration settings, the Charging Station will send Meter Values. |
| Purpose | To verify if the CSMS is able to handle a Charging Station sending clock-aligned Meter Values, when there is no ongoing transaction. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <p>1. The Test System notifies the CSMS about its measured Meter Values.</p> <p>Message: MeterValuesRequest</p> <ul style="list-style-type: none"> - timestamp <The intervals between the timestamps of the received Meter Value messages equals the _<Configured clock-aligned Meter Values interval>. - sampledValue.context is <i>Sample.Clock</i> <p>Message: NotifyEventRequest</p> <ul style="list-style-type: none"> - timestamp <The intervals between the timestamps of the received Meter Value messages equals the _<Configured clock-aligned Meter Values interval>. - trigger is <i>Periodic</i> - component.name is <i>FiscalMetering</i> <p><u>Note(s):</u></p> <ul style="list-style-type: none"> - This step will be executed every _<Configured clock-aligned Meter Values interval> - This step will be executed for <i>evseld=0</i> and all configured EVSE. - The Test System will end the testcase after it has send three Meter Value messages. | <p>2. The CSMS responds accordingly.</p> |

| |
|-----------------------------------|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_J_02_CSMS: Clock-aligned Meter Values - Transaction ongoing

| | |
|-------------------|--|
| Test case name | Clock-aligned Meter Values - Transaction ongoing |
| Test case Id | TC_J_02_CSMS |
| Use case Id(s) | J01 |
| Requirement(s) | J01.FR.18 |
| System under test | CSMS |
| Description | The Charging Station samples the electrical meter or other sensor/transducer hardware to provide information about its Meter Values. Depending on configuration settings, the Charging Station will send Meter Values. |
| Purpose | To verify if the CSMS is able to handle a Charging Station sending clock-aligned Meter Values, when there is an ongoing transaction. |
| Prerequisite(s) | N/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> for <Configured evseld> |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <p>1. The Test System notifies the CSMS about its measured Meter Values.</p> <p>Message: MeterValuesRequest</p> <ul style="list-style-type: none"> - timestamp <The intervals between the timestamps of the received Meter Value messages equals the configured clock-aligned Meter Values interval>. - sampledValue.context is <i>Sample.Clock</i> <p>Message: NotifyEventRequest</p> <ul style="list-style-type: none"> - timestamp <The intervals between the timestamps of the received Meter Value messages equals the configured clock-aligned Meter Values interval>. - trigger is <i>Periodic</i> - component.name is <i>FiscalMetering</i> <p><u>Note(s):</u></p> <ul style="list-style-type: none"> - This step will be executed every _<Configured clock-aligned Meter Values interval> - This step will be executed for evseld=0 and all configured idle EVSE. | <p>2. The CSMS responds accordingly.</p> |

| Main (Test scenario) | |
|--|---|
| <p>3. The Test System sends a TransactionEventRequest With triggerReason is <i>MeterValueClock</i> eventType is <i>Updated</i> timestamp <i><The intervals between the timestamps of the received Meter Value messages equals the configured clock-aligned Meter Values interval></i>. sampledValue.context is <i>Sample.Clock</i></p> <p><u>Note(s):</u> - <i>This step will be executed every _<Configured clock-aligned Meter Values interval></i> - <i>The Test System will end the testcase after the _<Configured transaction duration> is reached._</i></p> | <p>4. The CSMS responds with a TransactionEventResponse</p> |
| Tool validations | |
| N/a | |
| Post scenario validations: | |
| N/a | |

TC_J_03_CSMS: Clock-aligned Meter Values - EventType Ended

| | |
|-------------------|--|
| Test case name | Clock-aligned Meter Values - EventType Ended |
| Test case Id | TC_J_03_CSMS |
| Use case Id(s) | J01 |
| Requirement(s) | J01.FR.18 |
| System under test | CSMS |
| Description | The Charging Station samples the electrical meter or other sensor/transducer hardware to provide information about its Meter Values. Depending on configuration settings, the Charging Station will send Meter Values. |
| Purpose | To verify if the CSMS is able to handle a Charging Station sending clock-aligned Meter Values, when a transaction ends. |
| Prerequisite(s) | N/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> |

| | |
|--|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <p>1. Execute Reusable State <i>EVDIsconnected</i></p> <p>- The TransactionEventRequest containing eventType <i>Ended</i> contains the MeterValue field.</p> <p>- timestamp <i><The intervals between the timestamps of the received Meter Value messages equals the configured value at configured clock-aligned Tx ended Meter Values interval></i>.</p> <p>- sampledValue.context is <i>Sample.Clock</i> AND the last one has <i>Transaction.End</i></p> <p><u>Note(s):</u></p> <p>- <i>This step will be executed after the _<Configured transaction duration> is reached._</i></p> <p>- <i>This causes the transaction to stop.</i></p> | |

| |
|----------------------------|
| Tool validations |
| N/a |
| Post scenario validations: |
| N/a |

TC_J_04_CSMS: Clock-aligned Meter Values - Signed

| | |
|-------------------|--|
| Test case name | Clock-aligned Meter Values - Signed |
| Test case Id | TC_J_04_CSMS |
| Use case Id(s) | J01 |
| Requirement(s) | J01.FR.21 |
| System under test | CSMS |
| Description | The Charging Station samples the electrical meter or other sensor/transducer hardware to provide information about its Meter Values. Depending on configuration settings, the Charging Station will send Meter Values. |
| Purpose | To verify if the CSMS is able to handle a Charging Station sending clock-aligned Meter Values, when a transaction ends. |
| Prerequisite(s) | N/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> |

| | |
|---|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <p>1. Execute Reusable State <i>EVDIsconnected</i></p> <ul style="list-style-type: none">- The TransactionEventRequest containing eventType <i>Ended</i> contains the MeterValue field.- timestamp <i><The intervals between the timestamps of the received Meter Value messages equals the configured value at configured clock-aligned Tx ended Meter Values interval></i>.- sampledValue.context is <i>Sample.Clock</i> AND the last one has <i>Transaction.End</i>- sampledValue.signedMeterValue is <i><Generated SignedMeterValueType></i> <p><u>Note(s):</u></p> <ul style="list-style-type: none">- <i>This step will be executed after the _<Configured transaction duration> is reached._</i>- <i>This causes the transaction to stop.</i> | |

| |
|-----------------------------------|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_J_07_CSMS: Sampled Meter Values - EventType Started - EVSE known

| | |
|-------------------|--|
| Test case name | Sampled Meter Values - EventType Started - EVSE known |
| Test case Id | TC_J_07_CSMS |
| Use case Id(s) | J02 |
| Requirement(s) | J02.FR.19 |
| System under test | CSMS |
| Description | The Charging Station samples the electrical meter or other sensor/transducer hardware to provide information about its Meter Values. Depending on configuration settings, the Charging Station will send Meter Values. |
| Purpose | To verify if the CSMS is able to handle a Charging Station sending start sampled Meter Values, when a transaction starts. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <p>1. Execute Reusable State <i>EVConnectedPreSession</i></p> <p>- The TransactionEventRequest contains the MeterValue field.</p> <p>- sampledValue.context is <i>Transaction.Begin</i></p> | |

| |
|-----------------------------------|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_J_08_CSMS: Sampled Meter Values - Context Transaction.Begin - EVSE not known

| | |
|-------------------|--|
| Test case name | Sampled Meter Values - Context Transaction.Begin - EVSE not known |
| Test case Id | TC_J_08_CSMS |
| Use case Id(s) | J02 |
| Requirement(s) | J02.FR.19 |
| System under test | CSMS |
| Description | The Charging Station samples the electrical meter or other sensor/transducer hardware to provide information about its Meter Values. Depending on configuration settings, the Charging Station will send Meter Values. |
| Purpose | To verify if the CSMS is able to handle a Charging Station sending start sampled Meter Values, when a transaction starts. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>Authorized</i> | |
| 2. Execute Reusable State <i>EVConnectedPreSession</i> | |
| <div>- The TransactionEventRequest contains the MeterValue field.</div> <div>- sampledValue.context is <i>Transaction.Begin</i></div> | |
| 3. Execute Reusable State <i>EnergyTransferStarted</i> | |

| |
|-----------------------------------|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_J_09_CSMS: Sampled Meter Values - EventType Updated

| | |
|-------------------|--|
| Test case name | Sampled Meter Values - EventType Updated |
| Test case Id | TC_J_09_CSMS |
| Use case Id(s) | J02 |
| Requirement(s) | J02.FR.19 |
| System under test | CSMS |
| Description | The Charging Station samples the electrical meter or other sensor/transducer hardware to provide information about its Meter Values. Depending on configuration settings, the Charging Station will send Meter Values. |
| Purpose | To verify if the CSMS is able to handle a Charging Station sending sampled Meter Values, when there is an ongoing transaction. |
| Prerequisite(s) | N/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <p>1. The Test System sends a TransactionEventRequest With triggerReason is <i>MeterValuePeriodic</i> eventType is <i>Updated</i> timestamp <The intervals between the timestamps of the received Meter Value messages equals the configured sampled Meter Values interval>. sampledValue.context is <i>Sample.Periodic</i></p> <p><u>Note(s):</u> - This step will be executed every _<Configured sampled Meter Values interval> - The Test System will end the testcase after three MeterValues.</p> | <p>2. The CSMS responds with a TransactionEventResponse</p> |

| |
|----------------------------|
| Tool validations |
| N/a |
| Post scenario validations: |
| N/a |

TC_J_10_CSMS: Sampled Meter Values - EventType Ended

| Test case name | Sampled Meter Values - EventType Ended |
|-------------------|--|
| Test case Id | TC_J_10_CSMS |
| Use case Id(s) | J02 |
| Requirement(s) | J02.FR.19 |
| System under test | CSMS |
| Description | The Charging Station samples the electrical meter or other sensor/transducer hardware to provide information about its Meter Values. Depending on configuration settings, the Charging Station will send Meter Values. |
| Purpose | To verify if the CSMS is able to handle a Charging Station sending sampled Meter Values, when a transaction ends. |
| Prerequisite(s) | N/a |

| Before (Preparations) |
|--|
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> |

| Main (Test scenario) | |
|---|------|
| Charging Station | CSMS |
| <p>1. Execute Reusable State <i>EVDIsconnected</i></p> <p>- The TransactionEventRequest containing eventType <i>Ended</i> contains the MeterValue field.</p> <p>- timestamp <i><The intervals between the timestamps of the received Meter Value messages equals the configured value at configured clock-aligned Tx ended Meter Values interval></i>.</p> <p>- sampledValue.context is <i>Sample.Periodic</i> AND the last one has <i>Transaction.End</i></p> <p><u>Note(s):</u></p> <p>- <i>This step will be executed after the _<Configured transaction duration> is reached._</i></p> <p>- <i>This causes the transaction to stop.</i></p> | |

| Tool validations |
|--|
| N/a |
| Post scenario validations: N/a |

TC_J_11_CSMS: Sampled Meter Values - Signed

| | |
|-------------------|--|
| Test case name | Sampled Meter Values - Signed |
| Test case Id | TC_J_11_CSMS |
| Use case Id(s) | J02 |
| Requirement(s) | J02.FR.21 |
| System under test | CSMS |
| Description | The Charging Station samples the electrical meter or other sensor/transducer hardware to provide information about its Meter Values. Depending on configuration settings, the Charging Station will send Meter Values. |
| Purpose | To verify if the CSMS is able to handle a Charging Station sending sampled Meter Values, when a transaction ends. |
| Prerequisite(s) | N/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> |

| | |
|--|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <p>1. Execute Reusable State <i>EVDIsconnected</i></p> <ul style="list-style-type: none">- The TransactionEventRequest containing eventType <i>Ended</i> contains the MeterValue field.- timestamp <i><The intervals between the timestamps of the received Meter Value messages equals the configured value at configured clock-aligned Tx ended Meter Values interval></i>.- sampledValue.context is <i>Sample.Periodic</i> AND the last one has <i>Transaction.End</i>- sampledValue.signedMeterValue is <i><Generated SignedMeterValueType></i> <p><u>Note(s):</u></p> <ul style="list-style-type: none">- <i>This step will be executed after the _<Configured transaction duration> is reached._</i>- <i>This causes the transaction to stop.</i> | |

| |
|-----------------------------------|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

K Smart Charging

TC_K_01_CSMS: Set Charging Profile - TxDefaultProfile - Specific EVSE

| | |
|-------------------|--|
| Test case name | Set Charging Profile - TxDefaultProfile - Specific EVSE |
| Test case Id | TC_K_01_CSMS |
| Use case Id(s) | K01 |
| Requirement(s) | K01.FR.31 |
| System under test | CSMS |
| Description | To enable the CSMS to influence the charging power or current drawn from a specific EVSE or the entire Charging Station over a period of time. The CSMS sends a SetChargingProfileRequest to the Charging Station to influence the power or current drawn by EVs. The CSMS calculates a ChargingSchedule to stay within certain limits, which MAY be imposed by any external system. |
| Purpose | To verify if the CSMS is able to send a TxDefaultProfile charging profile for a specific EVSE as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| Main (Test scenario) | |
|---|--|
| Charging Station | CSMS |
| 2. The Test System responds with a SetChargingProfileResponse with status <i>Accepted</i> | 1. The CSMS sends a SetChargingProfileRequest with - chargingProfile.id <Configured chargingProfileId> |

| |
|--|
| Tool validations |
| <p>* Step 1:</p> <p>Message SetChargingProfileRequest</p> <p>evseld <Configured evseld> AND</p> <p>chargingProfile.stackLevel <Configured stackLevel> AND</p> <p>chargingProfile.chargingProfilePurpose TxDefaultProfile AND</p> <p>chargingProfile.chargingProfileKind Absolute AND</p> <p>chargingProfile.validFrom now AND</p> <p>chargingProfile.validTo now + <Configured Charging Schedule Duration> AND</p> <p>chargingProfile.chargingSchedule.startSchedule now AND</p> <p>chargingProfile.chargingSchedule.chargingRateUnit <Configured chargingRateUnit> AND</p> <p>chargingProfile.chargingSchedule.duration <Configured duration> AND</p> <p>chargingProfile.chargingSchedule.chargingSchedulePeriod.startPeriod 0 AND</p> <p>chargingProfile.chargingSchedule.chargingSchedulePeriod.limit 6.0 or 6000.0 AND</p> <p>chargingProfile.chargingSchedule.chargingSchedulePeriod.numberPhases <Configured numberPhases> where <Configured numberPhases> not 3 OR</p> <p>chargingProfile.chargingSchedule.chargingSchedulePeriod.numberPhases <Configured numberPhases> or <omit> where <Configured numberPhases> 3</p> <p>Post scenario validations:</p> <p>- N/a</p> |

TC_K_02_CSMS: Set Charging Profile - TxProfile without ongoing transaction on the specified EVSE

| | |
|-------------------|--|
| Test case name | Set Charging Profile - TxProfile without ongoing transaction on the specified EVSE |
| Test case Id | TC_K_02_CSMS |
| Use case Id(s) | K01 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | To enable the CSMS to influence the charging power or current drawn from a specific EVSE or the entire Charging Station over a period of time. The CSMS sends a SetChargingProfileRequest to the Charging Station to influence the power or current drawn by EVs. The CSMS calculates a ChargingSchedule to stay within certain limits, which MAY be imposed by any external system. |
| Purpose | To verify if the CSMS is able to send a TxProfile and read the charger's feedback while no transaction is ongoing for a specific EVSE as described at the OCPP specification. |
| Prerequisite(s) | If the CSMS supports sending a TxProfile while there is no transaction ongoing. |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a SetChargingProfileResponse with status <i>Rejected</i> | 1. The CSMS sends a SetChargingProfileRequest - chargingProfile.id < <i>Configured chargingProfileId</i> > |

| |
|--|
| Tool validations |
| <p>* Step 1:</p> <p>Message SetChargingProfileRequest</p> <ul style="list-style-type: none"> - evseld <Configured evseld> AND - chargingProfile.chargingProfilePurpose TxProfile AND - chargingProfile.stackLevel <Configured stackLevel> AND - chargingProfile.chargingProfileKind Relative AND - chargingProfile.chargingSchedule.startSchedule must be omitted AND - chargingProfile.chargingSchedule.chargingRateUnit <Configured chargingRateUnit> AND - chargingProfile.chargingSchedule.chargingSchedulePeriod.startPeriod 0 AND - chargingProfile.chargingSchedule.chargingSchedulePeriod.limit 7.0 or 7000.0 AND - chargingProfile.chargingSchedule.chargingSchedulePeriod.numberPhases <Configured numberPhases> where <Configured numberPhases> not 3 OR - chargingProfile.chargingSchedule.chargingSchedulePeriod.numberPhases <Configured numberPhases> or <omit> where <Configured numberPhases> 3 <p>Post scenario validations:</p> <ul style="list-style-type: none"> - N/a |

TC_K_03_CSMS: Set Charging Profile - ChargingStationMaxProfile

| | |
|-------------------|--|
| Test case name | Set Charging Profile - ChargingStationMaxProfile |
| Test case Id | TC_K_03_CSMS |
| Use case Id(s) | K01 |
| Requirement(s) | K01.FR.31, K01.FR.38 |
| System under test | CSMS |
| Description | To enable the CSMS to influence the charging power or current drawn from a specific EVSE or the entire Charging Station over a period of time. The CSMS sends a SetChargingProfileRequest to the Charging Station to influence the power or current drawn by EVs. The CSMS calculates a ChargingSchedule to stay within certain limits, which MAY be imposed by any external system. |
| Purpose | To verify if the CSMS is able to send a ChargingStationMaxProfile charging profile as described at the OCPP specification. |
| Prerequisite(s) | n/a |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State:

N/a

Main (Test scenario)

| Charging Station | CSMS |
|--|--|
| 2. The Test System responds with a SetChargingProfileResponse with status Accepted | 1. The CSMS sends a SetChargingProfileRequest - chargingProfile.id <Configured chargingProfileId> |

Tool validations

* Step 1:

Message **SetChargingProfileRequest****evseld** 0 AND**chargingProfile.stackLevel** <Configured stackLevel> AND**chargingProfile.chargingProfilePurpose** *ChargingStationMaxProfile* AND**chargingProfile.chargingProfileKind** *Absolute***chargingProfile.chargingSchedule.chargingRateUnit** <Configured ChargingRateUnit>**chargingProfile.chargingSchedule.duration** <Configured duration>**chargingProfile.chargingSchedule.chargingSchedulePeriod.startPeriod** 0**chargingProfile.chargingSchedule.chargingSchedulePeriod.limit** 8.0 or 8000.0**chargingProfile.chargingSchedule.chargingSchedulePeriod.numberPhases** <Configured numberPhases> where <Configured numberPhases> not 3 OR**chargingProfile.chargingSchedule.chargingSchedulePeriod.numberPhases** <Configured numberPhases> or <omit> where <Configured numberPhases> 3**chargingProfile.chargingSchedule.startSchedule** <Not omitted>

Post scenario validations:

- N/a

TC_K_04_CSMS: Replace charging profile - With chargingProfileId

| | |
|-------------------|--|
| Test case name | Replace charging profile - With chargingProfileId |
| Test case Id | TC_K_04_CSMS |
| Use case Id(s) | n/a |
| Requirement(s) | n/a |
| System under test | CSMS |
| Description | To enable the CSMS to influence the charging power or current drawn from a specific EVSE or the entire Charging Station over a period of time. The CSMS sends a SetChargingProfileRequest to the Charging Station to influence the power or current drawn by EVs. The CSMS calculates a ChargingSchedule to stay within certain limits, which MAY be imposed by any external system. |
| Purpose | To verify if the CSMS is able to replace a charging profile with the same ProfileKind, Purpose, and stackLevel, but a different limit. |
| Prerequisite(s) | n/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a SetChargingProfileResponse with status Accepted | 1. The CSMS sends a SetChargingProfileRequest with chargingProfile.chargingSchedule.chargingSchedulePeriod.limit 8.0 or 8000.0 |
| 4. The Test System responds with a SetChargingProfileResponse with status Accepted | 3. The CSMS sends a SetChargingProfileRequest with chargingProfile.chargingSchedule.chargingSchedulePeriod.limit 6.0 or 6000.0 |

| Tool validations |
|---|
| <p>* Step 1:</p> <p>Message SetChargingProfileRequest</p> <p>chargingProfile.chargingSchedule.chargingSchedulePeriod.startPeriod 0 AND</p> <p>chargingProfile.chargingSchedule.chargingSchedulePeriod.limit 8.0 (A) or 8000.0 (W)</p> <p>The chargingSchedule contains only one chargingSchedulePeriod</p> <p>* Step 3:</p> <p>Message SetChargingProfileRequest</p> <p>chargingProfile.chargingSchedule.chargingSchedulePeriod.startPeriod 0 AND</p> <p>chargingProfile.chargingSchedule.chargingSchedulePeriod.limit 6.0 (A) or 6000.0 (W)</p> <p>The chargingSchedule contains only one chargingSchedulePeriod</p> <p>* Step 1/3:</p> <p>Message SetChargingProfileRequest</p> <p>chargingProfile.id <Same id for both chargingProfiles></p> <p>chargingProfile.chargingSchedule.startSchedule must NOT be omitted.</p> <p>chargingProfile.stackLevel <Configured stackLevel> AND</p> <p>chargingProfile.chargingProfilePurpose <Equal value for both profiles> AND (TxDefaultProfile OR ChargingStationMaxProfile)</p> <p>chargingProfile.chargingProfileKind <Equal value for both profiles> AND</p> <p>If chargingProfile.chargingProfilePurpose is TxDefaultProfile then chargingProfile.chargingProfileKind must be Absolute OR Recurring</p> <p>If chargingProfile.chargingProfilePurpose is ChargingStationMaxProfile then chargingProfile.chargingProfileKind must be Absolute</p> <p>If chargingProfile.chargingProfileKind is Recurring then chargingProfile.recurrencyKind must NOT be omitted, else omitted</p> <p>The received Charging Profiles must comply with the requirements defined at part 2 specification.</p> |
| <p>Post scenario validations:</p> <p>- N/a</p> |

TC_K_05_CSMS: Clear Charging Profile - With chargingProfileId

| | |
|-------------------|--|
| Test case name | Clear Charging Profile - With chargingProfileId |
| Test case Id | TC_K_05_CSMS |
| Use case Id(s) | K10 |
| Requirement(s) | K10.FR.02 |
| System under test | CSMS |
| Description | If the CSMS wishes to clear some or all of the charging profiles that were previously sent to the Charging Station, then the CSMS sends a ClearChargingProfileRequest to the Charging Station. |
| Purpose | To verify if the CSMS is able to request the charging station to clear a specific charging profile (not TxDefault) with only a chargingProfileId as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: CSMS sends a GetChargingProfilesRequest Test System responds with a GetChargingProfilesResponse with status <i>Accepted</i> Test System sends a ReportChargingProfilesRequest CSMS responds with a ReportChargingProfilesResponse |
| Reusable State: N/a |

| Main (Test scenario) | |
|---|--|
| Charging Station | CSMS |
| 2. The Test System responds with a ClearChargingProfileResponse with status <i>Accepted</i> | 1. The CSMS sends a ClearChargingProfileRequest with chargingProfileId <i><Generated chargingProfileId></i> AND chargingProfileCriteria <i>omit</i> |

| |
|---|
| Tool validations |
| * Step 1: Message ClearChargingProfileRequest chargingProfileId <Generated chargingProfileId> AND chargingProfileCriteria omit |
| Post scenario validations: - N/a |

TC_K_06_CSMS: Clear Charging Profile - With stackLevel/purpose combination for one profile

| | |
|-------------------|--|
| Test case name | Clear Charging Profile - With stackLevel/purpose combination for one profile |
| Test case Id | TC_K_06_CSMS |
| Use case Id(s) | K10 |
| Requirement(s) | K10.FR.02 |
| System under test | CSMS |
| Description | If the CSMS wishes to clear some or all of the charging profiles that were previously sent to the Charging Station, then the CSMS sends a ClearChargingProfileRequest to the Charging Station. |
| Purpose | To verify if the CSMS is able to request the charging station to clear a specific charging profile with a stackLevel/purpose combination for a chargingProfileId as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a ClearChargingProfileResponse with status <i>Accepted</i> | 1. The CSMS sends a ClearChargingProfileRequest with chargingProfilePurpose <i>TxDefaultProfile</i> AND evseld <i><Configured evseld></i> AND stackLevel <i><Configured stackLevel></i> |

| |
|---|
| Tool validations |
| * Step 1: Message ClearChargingProfileRequest chargingProfileCriteria.chargingProfilePurpose <i>TxDefaultProfile</i> AND chargingProfileCriteria.stackLevel <i><Configured stackLevel></i> AND chargingProfileCriteria.evseld <i><Configured evseld></i> AND chargingProfileId must be omitted. |
| Post scenario validations: - N/a |

TC_K_08_CSMS: Clear Charging Profile - Without previous charging profile

| | |
|-------------------|---|
| Test case name | Clear Charging Profile - Without previous charging profile |
| Test case Id | TC_K_08_CSMS |
| Use case Id(s) | K10 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | If the CSMS wishes to clear some or all of the charging profiles that were previously sent to the Charging Station, then the CSMS sends a ClearChargingProfileRequest to the Charging Station. |
| Purpose | To verify if the CSMS is able to request the charging station to clear a specific charging profile with a chargingProfileId and stackLevel /purpose combination while the Charging stations does not accept as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a ClearChargingProfileResponse with status <i>Unknown</i> | 1. The CSMS sends a ClearChargingProfileRequest with chargingProfilePurpose <i>TxDefaultProfile</i> AND evseld <i><Configured evseld></i> AND stackLevel <i><Configured stackLevel></i> |

| |
|--|
| Tool validations |
| <p>* Step 1:</p> <p>Message ClearChargingProfileRequest</p> <p>chargingProfileCriteria.chargingProfilePurpose <i>TxDefaultProfile</i> AND</p> <p>chargingProfileCriteria.stackLevel <i><Configured stackLevel></i> AND</p> <p>chargingProfileCriteria.evseld <i><Configured evseld></i> AND</p> <p>chargingProfileId must be omitted.</p> |
| <p>Post scenario validations:</p> <p>- N/a</p> |

TC_K_10_CSMS: Set Charging Profile - TxDefaultProfile - All EVSE

| | |
|-------------------|--|
| Test case name | Set Charging Profile - TxDefaultProfile - All EVSE |
| Test case Id | TC_K_10_CSMS |
| Use case Id(s) | K01 |
| Requirement(s) | K01.FR.31 |
| System under test | CSMS |
| Description | To enable the CSMS to influence the charging power or current drawn from a specific EVSE or the entire Charging Station over a period of time. The CSMS sends a SetChargingProfileRequest to the Charging Station to influence the power or current drawn by EVs. The CSMS calculates a ChargingSchedule to stay within certain limits, which MAY be imposed by any external system. |
| Purpose | To verify if the CSMS is able to send a TxDefaultProfile charging profile for all EVSE as described at the OCPP specification. |
| Prerequisite(s) | n/a |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State:

N/a

Main (Test scenario)

| Charging Station | CSMS |
|--|--|
| 2. The Test System responds with a SetChargingProfileResponse with status Accepted | 1. The CSMS sends a SetChargingProfileRequest with - chargingProfile.id <Configured chargingProfileId> |

Tool validations

* Step 1:

Message **SetChargingProfileRequest****evseld** 0 AND**chargingProfile.stackLevel** <Configured stackLevel> AND**chargingProfile.chargingProfilePurpose** TxDefaultProfile AND**chargingProfile.chargingProfileKind** Absolute AND**chargingProfile.chargingSchedule.startSchedule** <Not omitted> AND**chargingProfile.chargingSchedule.chargingRateUnit** <Configured ChargingRateUnit> AND**chargingProfile.chargingSchedule.chargingSchedulePeriod.startPeriod** 0 AND**chargingProfile.chargingSchedule.duration** <Configured duration>**chargingProfile.chargingSchedule.chargingSchedulePeriod.limit** 6.0 or 6000.0 AND**chargingProfile.chargingSchedule.chargingSchedulePeriod.numberPhases** <Configured numberPhases> where <Configured numberPhases> not 3 OR**chargingProfile.chargingSchedule.chargingSchedulePeriod.numberPhases** <Configured numberPhases> or <omit> where <Configured numberPhases> 3

Post scenario validations:

- N/a

TC_K_15_CSMS: Set Charging Profile - Not Supported

| | |
|-------------------|--|
| Test case name | Set Charging Profile - Not Supported |
| Test case Id | TC_K_15_CSMS |
| Use case Id(s) | K01 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | To enable the CSMS to influence the charging power or current drawn from a specific EVSE or the entire Charging Station over a period of time. The CSMS sends a SetChargingProfileRequest to the Charging Station to influence the power or current drawn by EVs. The CSMS calculates a ChargingSchedule to stay within certain limits, which MAY be imposed by any external system. |
| Purpose | To verify if the CSMS is able to send a Profile, while the charging station does not support chargingprofiles, and read the response as described at the OCPP specification. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with RPC Framework: CALLERROR: NotSupported. | <p>1. The CSMS sends a SetChargingProfileRequest with:</p> <p>evseld <Configured evseld> AND</p> <p>chargingProfile.stackLevel <Configured stackLevel> AND</p> <p>chargingProfile.chargingProfilePurpose TxDefaultProfile AND</p> <p>chargingProfile.chargingProfileKind Absolute AND</p> <p>chargingProfile.chargingSchedule.startSchedule <Not omitted> AND</p> <p>chargingProfile.chargingSchedule.chargingRateUnit <Configured ChargingRateUnit> AND</p> <p>chargingProfile.chargingSchedule.chargingSchedulePeriod.startPeriod 0 AND</p> <p>chargingProfile.chargingSchedule.duration <Configured duration></p> <p>chargingProfile.chargingSchedule.chargingSchedulePeriod.limit 6.0 or 6000.0 AND</p> <p>chargingProfile.chargingSchedule.chargingSchedulePeriod.numberPhases <Configured numberPhases> where <Configured numberPhases> not 3 OR</p> <p>chargingProfile.chargingSchedule.chargingSchedulePeriod.numberPhases <Configured numberPhases> or <omit> where <Configured numberPhases> 3</p> |

| Tool validations |
|---|
| <p>* Step 1:</p> <p>Message SetChargingProfileRequest</p> <p>evseld <Configured evseld> AND</p> <p>chargingProfile.stackLevel <Configured stackLevel> AND</p> <p>chargingProfile.chargingProfilePurpose <i>TxDefaultProfile</i> AND</p> <p>chargingProfile.chargingProfileKind <i>Absolute</i> AND</p> <p>chargingProfile.chargingSchedule.startSchedule <Not omitted> AND</p> <p>chargingProfile.chargingSchedule.chargingRateUnit <Configured ChargingRateUnit> AND</p> <p>chargingProfile.chargingSchedule.chargingSchedulePeriod.startPeriod 0 AND</p> <p>chargingProfile.chargingSchedule.duration <Configured duration></p> <p>chargingProfile.chargingSchedule.chargingSchedulePeriod.limit 6.0 or 6000.0 AND</p> <p>chargingProfile.chargingSchedule.chargingSchedulePeriod.numberPhases <Configured numberPhases></p> |
| <p>Post scenario validations:</p> <p>- N/a</p> |

TC_K_19_CSMS: Set Charging Profile - ChargingProfileKind is Recurring

| | |
|-------------------|--|
| Test case name | Set Charging Profile - ChargingProfileKind is Recurring |
| Test case Id | TC_K_19_CSMS |
| Use case Id(s) | K01 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | To enable the CSMS to influence the charging power or current drawn from a specific EVSE or the entire Charging Station over a period of time. The CSMS sends a SetChargingProfileRequest to the Charging Station to influence the power or current drawn by EVs. The CSMS calculates a ChargingSchedule to stay within certain limits, which MAY be imposed by any external system. |
| Purpose | To verify if the CSMS is able to send a Profile with a recurrencyKind specified as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a SetChargingProfileResponse with - status <i>Accepted</i> | 1. The CSMS sends a SetChargingProfileRequest |

| |
|---|
| Tool validations |
| <p>* Step 1:</p> <p>Message SetChargingProfileRequest</p> <ul style="list-style-type: none"> - evseld <Configured evseld> AND - chargingProfile.stackLevel <Configured stackLevel> AND - chargingProfile.chargingProfilePurpose TxDefaultProfile AND - chargingProfile.chargingSchedule.chargingSchedulePeriod.startPeriod 0 AND - chargingProfile.chargingProfileKind Recurring AND - chargingProfile.recurrencyKind <Configured recurrencyKind> |
| <p>Post scenario validations:</p> <ul style="list-style-type: none"> - N/a |

TC_K_29_CSMS: Get Charging Profile - Evseld 0

| | |
|-------------------|---|
| Test case name | Get Charging Profile - Evseld 0 |
| Test case Id | TC_K_29_CSMS |
| Use case Id(s) | K09 |
| Requirement(s) | K09.FR.03 |
| System under test | CSMS |
| Description | With the GetChargingProfilesRequest message the CSMS can ask a Charging Station to report all, or a subset of all the install Charging Profiles from the different possible sources. This can be used for some automatic smart charging control system, or for debug purposes by a CSO. |
| Purpose | To verify if the CSMS is able to request charging profiles installed on the charging station itself and read in the reports as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: <i>EnergyTransferStarted</i> |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a GetChargingProfilesResponse with - <i>status Accepted</i> | 1. The CSMS sends a GetChargingProfilesRequest with - <i>evseld 0</i> |
| 3. The Test System sends a ReportChargingProfilesRequest with - <i>requestId <Received requestId></i> | 4. The CSMS responds with a ReportChargingProfilesResponse |

| |
|---|
| Tool validations |
| <p>* Step 1:</p> <p>Message GetChargingProfilesRequest</p> <p>- <i>evseld 0</i> AND</p> <p>- <i>chargingProfile.chargingProfilePurpose <Configured chargingProfilePurpose></i> AND</p> <p><u>Note</u>: <i>chargingProfilePurpose</i> is included, because the <i>chargingProfile</i> field is required and may not be left empty.</p> <p>- <i>chargingProfile.stackLevel</i> must be omitted AND</p> <p>- <i>chargingProfile.chargingLimitSource</i> must be omitted AND</p> <p>- <i>chargingProfile.chargingProfileId</i> must be omitted</p> |
| Post scenario validations: |
| - N/a |

TC_K_30_CSMS: Get Charging Profile - Evseld > 0

| | |
|-------------------|---|
| Test case name | Get Charging Profile - Evseld > 0 |
| Test case Id | TC_K_30_CSMS |
| Use case Id(s) | K09 |
| Requirement(s) | K09.FR.03 |
| System under test | CSMS |
| Description | With the GetChargingProfilesRequest message the CSMS can ask a Charging Station to report all, or a subset of all the install Charging Profiles from the different possible sources. This can be used for some automatic smart charging control system, or for debug purposes by a CSO. |
| Purpose | To verify if the CSMS is able to request charging profiles installed on a specific EVSE and read in the reports as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| Main (Test scenario) | |
|--|---|
| Charging Station | CSMS |
| 2. The Test System responds with a GetChargingProfilesResponse with - status <i>Accepted</i> | 1. The CSMS sends a GetChargingProfilesRequest |
| 3. The Test System sends a ReportChargingProfilesRequest with - requestId <i><Received requestId></i> | 4. The CSMS responds with a ReportChargingProfilesResponse |

| |
|---|
| Tool validations |
| <p>* Step 1:</p> <p>Message GetChargingProfilesRequest</p> <ul style="list-style-type: none"> - evseld <i><Configured evseld></i> AND - chargingProfile.chargingProfilePurpose <i><Configured chargingProfilePurpose></i> AND <p><u>Note</u>: <i>chargingProfilePurpose</i> is included, because the <i>chargingProfile</i> field is required and may not be left empty.</p> <ul style="list-style-type: none"> - chargingProfile.stackLevel must be omitted AND - chargingProfile.chargingLimitSource must be omitted AND - chargingProfile.chargingProfileId must be omitted |
| Post scenario validations: |
| - N/a |

TC_K_31_CSMS: Get Charging Profile - No Evseld

| | |
|--------------------------|---|
| Test case name | Get Charging Profile - No Evseld |
| Test case Id | TC_K_31_CSMS |
| Use case Id(s) | K09 |
| Requirement(s) | K09.FR.03 |
| System under test | CSMS |
| Description | With the GetChargingProfilesRequest message the CSMS can ask a Charging Station to report all, or a subset of all the install Charging Profiles from the different possible sources. This can be used for some automatic smart charging control system, or for debug purposes by a CSO. |
| Purpose | To verify if the CSMS is able to request all charging profiles installed on a charger and read in the reports as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a GetChargingProfilesResponse with - status <i>Accepted</i> | 1. The CSMS sends a GetChargingProfilesRequest with - requestId <i><Received requestId></i> |
| 3. The Test System sends a ReportChargingProfilesRequest with - requestId <i><Received requestId></i> AND - tb <i>true</i> AND - evseld <i>i</i> | 4. The CSMS responds with a ReportChargingProfilesResponse |
| Note(s): - Step 3 and 4 are repeated for every evse | |

| |
|--|
| Tool validations |
| <p>* Step 1:</p> <p>Message GetChargingProfilesRequest</p> <ul style="list-style-type: none"> - evseld must be omitted AND - chargingProfile.chargingProfilePurpose <i><Configured chargingProfilePurpose></i> AND <p><i>Note: chargingProfilePurpose is included, because the chargingProfile field is required and may not be left empty.</i></p> <ul style="list-style-type: none"> - chargingProfile.stackLevel must be omitted AND - chargingProfile.chargingLimitSource must be omitted AND - chargingProfile.chargingProfileId must be omitted |
| <p>Post scenario validations:</p> <ul style="list-style-type: none"> - N/a |

TC_K_32_CSMS: Get Charging Profile - chargingProfileId

| | |
|-------------------|---|
| Test case name | Get Charging Profile - chargingProfileId |
| Test case Id | TC_K_32_CSMS |
| Use case Id(s) | K09 |
| Requirement(s) | K09.FR.03 |
| System under test | CSMS |
| Description | With the GetChargingProfilesRequest message the CSMS can ask a Charging Station to report all, or a subset of all the install Charging Profiles from the different possible sources. This can be used for some automatic smart charging control system, or for debug purposes by a CSO. |
| Purpose | To verify if the CSMS is able to request a specific charging profile and read in the reports as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a GetChargingProfilesResponse with - status <i>Accepted</i> | 1. The CSMS sends a GetChargingProfilesRequest - chargingProfileId <Received <i>chargingProfileId</i> > |
| 3. The Test System sends a ReportChargingProfilesRequest with - requestId <i>Generated Id</i> | 4. The CSMS responds with a ReportChargingProfilesResponse |

| |
|---|
| Tool validations |
| <p>* Step 1:</p> <p>Message GetChargingProfilesRequest</p> <ul style="list-style-type: none"> - evseld must be omitted AND - chargingProfile.chargingProfilePurpose must be omitted AND - chargingProfile.stackLevel must be omitted AND - chargingProfile.chargingLimitSource must be omitted AND - chargingProfile.chargingProfileId <received <i>chargingProfileId</i>> |
| <p>Post scenario validations:</p> <ul style="list-style-type: none"> - N/a |

TC_K_33_CSMS: Get Charging Profile - Evseld > 0 + stackLevel

| | |
|-------------------|---|
| Test case name | Get Charging Profile - Evseld > 0 + stackLevel |
| Test case Id | TC_K_33_CSMS |
| Use case Id(s) | K09 |
| Requirement(s) | K09.FR.03 |
| System under test | CSMS |
| Description | With the GetChargingProfilesRequest message the CSMS can ask a Charging Station to report all, or a subset of all the install Charging Profiles from the different possible sources. This can be used for some automatic smart charging control system, or for debug purposes by a CSO. |
| Purpose | To verify if the CSMS is able to request charging profiles with a specific stackLevel installed on a specific EVSE and read in the reports as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a GetChargingProfilesResponse with - status <i>Accepted</i> | 1. The CSMS sends a GetChargingProfilesRequest |
| 3. The Test System sends a ReportChargingProfilesRequest with - requestId <i>Generated Id</i> | 4. The CSMS responds with a ReportChargingProfilesResponse |

| |
|---|
| Tool validations |
| <p>* Step 1:</p> <p>Message GetChargingProfilesRequest</p> <ul style="list-style-type: none"> - evseld <Configured evseld> AND - chargingProfile.stackLevel <Configured stackLevel> AND - chargingProfile.chargingProfilePurpose must be omitted AND - chargingProfile.chargingLimitSource must be omitted AND - chargingProfile.chargingProfileId must be omitted |
| <p>Post scenario validations:</p> <ul style="list-style-type: none"> - N/a |

TC_K_34_CSMS: Get Charging Profile - Evseld > 0 + chargingLimitSource

| | |
|-------------------|---|
| Test case name | Get Charging Profile - Evseld > 0 + chargingLimitSource |
| Test case Id | TC_K_34_CSMS |
| Use case Id(s) | K09 |
| Requirement(s) | K09.FR.03 |
| System under test | CSMS |
| Description | With the GetChargingProfilesRequest message the CSMS can ask a Charging Station to report all, or a subset of all the install Charging Profiles from the different possible sources. This can be used for some automatic smart charging control system, or for debug purposes by a CSO. |
| Purpose | To verify if the CSMS is able to request charging profiles with a specific chargingLimitSource installed on a specific EVSE and read in the reports as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a GetChargingProfilesResponse with - status <i>Accepted</i> | 1. The CSMS sends a GetChargingProfilesRequest |
| 3. The Test System sends a ReportChargingProfilesRequest with - requestId <i>Generated Id</i> | 4. The CSMS responds with a ReportChargingProfilesResponse |

| |
|--|
| Tool validations |
| <p>* Step 1:</p> <p>Message GetChargingProfilesRequest</p> <ul style="list-style-type: none"> - evseld <Configured evseld> AND - chargingProfile.chargingLimitSource <Configured chargingLimitSource> AND - chargingProfile.stackLevel must be omitted AND - chargingProfile.chargingProfilePurpose must be omitted AND - chargingProfile.chargingProfileId must be omitted |
| Post scenario validations: |
| - N/a |

TC_K_35_CSMS: Get Charging Profile - Evseld > 0 + chargingProfilePurpose

| | |
|-------------------|---|
| Test case name | Get Charging Profile - Evseld > 0 + chargingProfilePurpose |
| Test case Id | TC_K_35_CSMS |
| Use case Id(s) | K09 |
| Requirement(s) | K09.FR.03 |
| System under test | CSMS |
| Description | With the GetChargingProfilesRequest message the CSMS can ask a Charging Station to report all, or a subset of all the install Charging Profiles from the different possible sources. This can be used for some automatic smart charging control system, or for debug purposes by a CSO. |
| Purpose | To verify if the CSMS is able to request charging profiles with a specific chargingProfilePurpose installed on a specific EVSE and read in the reports as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a GetChargingProfilesResponse with - status <i>Accepted</i> | 1. The CSMS sends a GetChargingProfilesRequest |
| 3. The Test System sends a ReportChargingProfilesRequest with - requestId <i>Generated Id</i> | 4. The CSMS responds with a ReportChargingProfilesResponse |

| |
|---|
| Tool validations |
| <p>* Step 1:</p> <p>Message GetChargingProfilesRequest</p> <ul style="list-style-type: none"> - evseld <Configured evseld> AND - chargingProfile.chargingProfilePurpose <Configured chargingProfilePurpose> AND - chargingProfile.stackLevel must be omitted AND - chargingProfile.chargingLimitSource must be omitted AND - chargingProfile.chargingProfileId must be omitted |
| Post scenario validations: |
| - N/a |

TC_K_36_CSMS: Get Charging Profile - Evseld > 0 + chargingProfilePurpose + stackLevel

| | |
|-------------------|---|
| Test case name | Get Charging Profile - Evseld > 0 + chargingProfilePurpose + stackLevel |
| Test case Id | TC_K_36_CSMS |
| Use case Id(s) | K09 |
| Requirement(s) | K09.FR.03 |
| System under test | CSMS |
| Description | With the GetChargingProfilesRequest message the CSMS can ask a Charging Station to report all, or a subset of all the install Charging Profiles from the different possible sources. This can be used for some automatic smart charging control system, or for debug purposes by a CSO. |
| Purpose | To verify if the CSMS is able to request charging profiles with a specific chargingProfilePurpose AND stackLevel installed on a specific EVSE and read in the reports as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| Main (Test scenario) | |
|--|---|
| Charging Station | CSMS |
| 2. The Test System responds with a GetChargingProfilesResponse with - status <i>Accepted</i> | 1. The CSMS sends a GetChargingProfilesRequest |
| 3. The Test System sends a ReportChargingProfilesRequest with - requestId <i>Generated Id</i> | 4. The CSMS responds with a ReportChargingProfilesResponse |

| |
|--|
| Tool validations |
| <p>* Step 1:</p> <p>Message GetChargingProfilesRequest</p> <ul style="list-style-type: none"> - evseld <Configured evseld> AND - chargingProfile.chargingProfilePurpose <Configured chargingProfilePurpose> AND - chargingProfile.chargingLimitSource must be omitted AND - chargingProfile.stackLevel <Configured stackLevel> AND - chargingProfile.chargingProfileId must be omitted |
| <p>Post scenario validations:</p> <ul style="list-style-type: none"> - N/a |

TC_K_60_CSMS: Set Charging Profile - TxProfile with ongoing transaction on the specified EVSE

| | |
|-------------------|---|
| Test case name | Set Charging Profile - TxProfile with ongoing transaction on the specified EVSE |
| Test case Id | TC_K_60_CSMS |
| Use case Id(s) | K01 |
| Requirement(s) | K01.FR.03, K01.FR.31 |
| System under test | CSMS |
| Description | The CSMS sets a TxProfile on a specific EVSE for a currently ongoing transaction. |
| Purpose | To verify if the CSMS is able to exchange messages to set a TxProfile on a specific EVSE for a currently ongoing transaction. |
| Prerequisite(s) | N/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a SetChargingProfileResponse With status is <i>Accepted</i> | 1. The CSMS sends a SetChargingProfileRequest |

| |
|---|
| Tool validations |
| <p>* Step 1: (Message: SetChargingProfileRequest) chargingProfile.chargingProfilePurpose is <i>TxProfile</i> AND chargingProfile.evseId is <i><Configured evseId></i> AND chargingProfile.transactionId <i><Generated transactionId></i> AND chargingProfile.chargingProfileKind is <i>Relative</i> OR <i>Absolute</i> If chargingProfileKind is <i>Relative</i> then chargingSchedule.startSchedule must be omitted. If chargingProfileKind is <i>Absolute</i> then chargingSchedule.startSchedule must NOT be omitted.</p> <p>The received Charging Profile must comply with the requirements defined at part 2 specification.</p> |
| Post scenario validations: N/a |

TC_K_37_CSMS: Remote start transaction with charging profile - Success

| | |
|-------------------|--|
| Test case name | Remote start transaction with charging profile - Success |
| Test case Id | TC_K_37_CSMS |
| Use case Id(s) | K05,F01 |
| Requirement(s) | K05.FR.02,F01.FR.08,F01.FR.09,F01.FR.11 |
| System under test | CSMS |
| Description | The CSMS sets a TxProfile on a specific EVSE inside a RequestStartTransactionRequest message. |
| Purpose | To verify if the CSMS is able to set a TxProfile on a specific EVSE in a RequestStartTransactionRequest message. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|--|---|
| Charging Station | CSMS |
| 2. The Test System responds with a RequestStartTransactionResponse With status <i>Accepted</i> | 1. The CSMS sends a RequestStartTransactionRequest |
| 3. The Test System sends a TransactionEventRequest With triggerReason <i>RemoteStart</i> transactionInfo.remoteStartId is present. | 4. The CSMS responds with a TransactionEventResponse |

| |
|---|
| Tool validations |
| <p>* Step 1:</p> <p>Message: RequestStartTransactionRequest</p> <p>idToken.idToken <Configured valid idToken></p> <p>idToken.type <Configured valid idToken type></p> <p>evseld <Configured evseld></p> <p>chargingProfile contains:</p> <p>chargingProfile.chargingProfilePurpose is <i>TxProfile</i></p> <p>chargingProfile.transactionId is omitted</p> <p>chargingProfile.chargingProfileKind is <i>Relative</i> OR <i>Absolute</i></p> <p>If chargingProfileKind is <i>Relative</i> then chargingSchedule.startSchedule must be omitted.</p> <p>If chargingProfileKind is <i>Absolute</i> then chargingSchedule.startSchedule must NOT be omitted.</p> <p>The received Charging Profile must comply with the requirements defined at part 2 specification.</p> |
| Post scenario validations: N/a |

TC_K_43_CSMS: Get Composite Schedule - Specific EVSE

| | |
|-------------------|---|
| Test case name | Get Composite Schedule - Specific EVSE |
| Test case Id | TC_K_43_CSMS |
| Use case Id(s) | K08 |
| Requirement(s) | K08.FR.01 |
| System under test | CSMS |
| Description | The CSMS requests a composite schedule which is a combination of local limits and the prevailing Charging Profiles of the different chargingProfilePurposes and stack levels. |
| Purpose | To verify if the CSMS is able to calculate request a composite schedule from the Charging Station for a specific EVSE. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <p>2. The Test System responds with a GetCompositeScheduleResponse With status Accepted schedule.evseId 1 schedule.duration is 300 schedule.chargingRateUnit <Specified chargingRateUnit from step 1> schedule.chargingSchedulePeriod[0].startPeriod 0 Note: Multiply limit by 1000 if chargingRateUnit is W schedule.chargingSchedulePeriod[0].limit 10</p> | <p>1. The CSMS sends a GetCompositeScheduleRequest</p> |

| |
|---|
| Tool validations |
| <p>* Step 1: (Message: GetCompositeScheduleRequest) evseId 1 duration is <Configured duration> chargingRateUnit <Configured chargingRateUnit></p> |
| <p>Post scenario validations: N/a</p> |

TC_K_44_CSMS: Get Composite Schedule - Charging Station

| | |
|-------------------|---|
| Test case name | Get Composite Schedule - Charging Station |
| Test case Id | TC_K_44_CSMS |
| Use case Id(s) | K08 |
| Requirement(s) | K08.FR.01 |
| System under test | CSMS |
| Description | The CSMS requests a composite schedule which is a combination of local limits and the prevailing Charging Profiles of the different chargingProfilePurposes and stack levels. |
| Purpose | To verify if the CSMS is able to calculate request a composite schedule from the Charging Station. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|---|---|
| Charging Station | CSMS |
| <p>2. The Test System responds with a GetCompositeScheduleResponse</p> <p>With status <i>Accepted</i></p> <p>schedule.evseId 0</p> <p>schedule.duration is 300</p> <p>schedule.chargingRateUnit <Specified <i>chargingRateUnit</i> from step 1></p> <p>schedule.chargingSchedulePeriod[0].startPeriod 0</p> <p>Note: Multiply limit by 1000 if <i>chargingRateUnit</i> is W</p> <p>schedule.chargingSchedulePeriod[0].limit 10</p> | <p>1. The CSMS sends a GetCompositeScheduleRequest</p> |

| |
|---|
| Tool validations |
| <p>* Step 1: (Message: GetCompositeScheduleRequest) evseId 0 duration is <Configured duration> chargingRateUnit <Configured chargingRateUnit></p> |
| <p>Post scenario validations: N/a</p> |

TC_K_48_CSMS: EMS Control - Set / Update External Charging Limit (not on a transaction)

| | |
|-------------------|--|
| Test case name | EMS Control - Set / Update External Charging Limit (not on a transaction) |
| Test case Id | TC_K_48_CSMS |
| Use case Id(s) | K12 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | A charging schedule or charging limit can be imposed by an external system on the Charging Station for new transactions or on the grid connection. An External Control System sends a charging limit to a Charging Station. This limit is then sent to the CSMS. |
| Purpose | To verify if the CSMS is able to receive the request from a charging station and respond correctly as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Test System sends a NotifyChargingLimitRequest with - chargingLimit.chargingLimitSource <i>EMS</i> | 2. The CSMS responds with a NotifyChargingLimitResponse |

| |
|-------------------------------------|
| Tool validations |
| - N/a |
| Post scenario validations: - N/a |

TC_K_50_CSMS: EMS Control - Reset / release external charging limit - Without ongoing transaction

| | |
|-------------------|--|
| Test case name | EMS Control - Reset / release external charging limit - Without ongoing transaction |
| Test case Id | TC_K_50_CSMS |
| Use case Id(s) | K13 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | A charging schedule or charging limit can be removed by an external system on the Charging Station. An external control system sends a signal to release a previously imposed charging limit to a Charging Station. The Charging Station notifies the CSMS about this. |
| Purpose | To verify if the CSMS is able to receive the notify from a charging station and respond correctly as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Test System sends a ClearedChargingLimitRequest with - chargingLimitSource <i>EMS</i> | 2. The CSMS responds with a ClearedChargingLimitResponse |

| |
|-------------------------------------|
| Tool validations |
| - N/a |
| Post scenario validations: - N/a |

TC_K_51_CSMS: EMS Control - Reset / release external charging limit - With ongoing transaction

| | |
|-------------------|--|
| Test case name | EMS Control - Reset / release external charging limit - With ongoing transaction |
| Test case Id | TC_K_51_CSMS |
| Use case Id(s) | K13 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | A charging schedule or charging limit can be removed by an external system on the Charging Station. An external control system sends a signal to release a previously imposed charging limit to a Charging Station. The Charging Station notifies the CSMS about this. |
| Purpose | To verify if the CSMS is able to receive the notify from a charging station and respond correctly as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Test System sends a ClearedChargingLimitRequest with - chargingLimitSource <i>EMS</i> | 2. The CSMS responds with a ClearedChargingLimitResponse |
| 3. The Test System sends a TransactionEventRequest with - eventType <i>Updated</i> - triggerReason <i>ChargingRateChanged</i> | 4. The CSMS responds with a TransactionEventResponse |

| |
|-------------------------------------|
| Tool validations |
| - N/a |
| Post scenario validations: - N/a |

TC_K_52_CSMS: EMS Control - Set / Update External Charging Limit (not on a transaction) - ChargingStationExternalConstraints in report

| | |
|-------------------|--|
| Test case name | EMS Control - Set / Update External Charging Limit (not on a transaction) - ChargingStationExternalConstraints in report |
| Test case Id | TC_K_52_CSMS |
| Use case Id(s) | K12 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | A charging schedule or charging limit can be removed by an external system on the Charging Station. An external control system sends a signal to release a previously imposed charging limit to a Charging Station. The Charging Station notifies the CSMS about this. |
| Purpose | To verify if the CSMS is able to correctly receive the report when a charging limit has been externally changed in a charging station as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a GetChargingProfilesResponse with - status <i>Accepted</i> | 1. The CSMS sends a GetChargingProfilesRequest |
| 3. The Test System sends a ReportChargingProfilesRequest with - requestId <i>Generated Id</i> - chargingProfile.chargingProfilePurpose <i>ChargingStationExternalConstraints</i> | 4. The CSMS responds with a ReportChargingProfilesResponse |

| |
|-------------------------------------|
| Tool validations |
| N/a |
| Post scenario validations: - N/a |

TC_K_53_CSMS: Charging with load leveling based on High Level Communication - Success

| | |
|-------------------|--|
| Test case name | Charging with load leveling based on High Level Communication - Success |
| Test case Id | TC_K_53_CSMS |
| Use case Id(s) | K15 |
| Requirement(s) | K15.FR.02,K15.FR.03,K15.FR.05,K15.FR.07,K15.FR.11 |
| System under test | CSMS |
| Description | ISO15118-1 E1 AC Charging with load leveling based on High Level Communication, and E4 DC charging with load leveling based on High Level Communication. |
| Purpose | To verify if the CSMS is able to perform load leveling when it receives the EV charging needs from the Charging Station. |
| Prerequisite(s) | N/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>Authorized</i> State is <i>EVConnectedPreSession</i> |

| | |
|---|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute reusable state <i>ISO15118SmartCharging</i> | |
| 2. The CSMS does NOT send a SetChargingProfileRequest | |
| <u>Note(s):</u> - The CSMS must NOT initiate a renegotiate after starting the transaction, without cause. For example; a smart charging algorithm or an external trigger, etc. | |

| |
|-----------------------------------|
| Tool validations |
| - N/a |
| Post scenario validations: N/a |

TC_K_55_CSMS: Charging with load leveling based on High Level Communication - EV charging profile exceeds limits

| | |
|-------------------|--|
| Test case name | Charging with load leveling based on High Level Communication - EV charging profile exceeds limits |
| Test case Id | TC_K_55_CSMS |
| Use case Id(s) | K15,K16,K17 |
| Requirement(s) | K15.FR.12,K15.FR.13,K16.FR.07,K16.FR.08,K17.FR.12,K17.FR.13 |
| System under test | CSMS |
| Description | ISO15118-1 E1 AC Charging with load leveling based on High Level Communication, and E4 DC charging with load leveling based on High Level Communication. |
| Purpose | To verify if the CSMS is able to renegotiate when it receives the EV charging schedule which exceeds the profile limits. |
| Prerequisite(s) | N/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>Authorized</i> State is <i>EVConnectedPreSession</i> |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Test System sends a NotifyEVChargingNeedsRequest . With evseld <Configured evseld> maxScheduleTuples & chargingNeeds <Configured values from mock EV> | 2. The CSMS responds with a NotifyEVChargingNeedsResponse . |
| 4. The Test System responds with a SetChargingProfileResponse With status <i>Accepted</i> | 3. The CSMS sends a SetChargingProfileRequest |
| 5. The Test System sends a NotifyEVChargingScheduleRequest . With evseld <Configured evseld> chargingSchedule <ChargingSchedule that exceeds the limits of the chargingSchedule provided at step 3.> | 6. The CSMS responds with a NotifyEVChargingScheduleResponse . |
| 7. The Test System sends a TransactionEventRequest . With triggerReason <i>ChargingStateChanged</i> transactionInfo.chargingState <i>Charging</i> | 8. The CSMS responds with a TransactionEventResponse . |
| 10. The Test System responds with a SetChargingProfileResponse With status <i>Accepted</i> | 9. The CSMS sends a SetChargingProfileRequest |
| 11. The Test System sends a NotifyEVChargingScheduleRequest . With evseld <Configured evseld> chargingSchedule <ChargingSchedule provided at step 9> | 12. The CSMS responds with a NotifyEVChargingScheduleResponse . |

| Tool validations |
|--|
| <div><div>* Step 2: (Message: NotifyEVChargingNeedsResponse) status <i>Accepted</i></div><div>* Step 3: (Message: SetChargingProfileRequest) evseld <i><Configured evseld></i> chargingProfilePurpose <i>TxProfile</i> transactionId <i><Provided transactionId from before></i></div><div>* Step 6: (Message: NotifyEVChargingScheduleResponse) status <i>Rejected</i></div><div>* Step 9: (Message: SetChargingProfileRequest) evseld <i><Configured evseld></i> chargingProfilePurpose <i>TxProfile</i> transactionId <i><Provided transactionId from before></i></div><div>* Step 12: (Message: NotifyEVChargingScheduleResponse) status <i>Accepted</i></div></div> |
| <div>Post scenario validations: N/a</div> |

TC_K_57_CSMS: Renegotiating a Charging Schedule - Initiated by EV

| | |
|-------------------|---|
| Test case name | Renegotiating a Charging Schedule - Initiated by EV |
| Test case Id | TC_K_57_CSMS |
| Use case Id(s) | K17 |
| Requirement(s) | K17.FR.02,K17.FR.03,K17.FR.05,K17.FR.07,K17.FR.11 |
| System under test | CSMS |
| Description | The EV signals the Charging Station that it wants to renegotiate and it provides new charging needs, which the Charging Station sends to the CSMS. Based on this and other parameters, the CSMS calculates a new charging schedule and sends it via SetChargingProfileRequest to Charging Station, which communicates it to the EV. |
| Purpose | To verify if the CSMS is able to renegotiate when the EV signals the Charging Station that it wants to renegotiate and it provides new charging needs, which the Charging Station sends to the CSMS. |
| Prerequisite(s) | N/a |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State(s):

State is

[Authorized](#) AND[EVConnectedPreSession](#) AND[ISO15118SmartCharging](#)

Main (Test scenario)

| Charging Station | CSMS |
|--|---|
| 1. The Test System sends a NotifyEVChargingNeedsRequest . With evseld <Configured evseld> maxScheduleTuples & chargingNeeds <Configured values from mock EV> | 2. The CSMS responds with a NotifyEVChargingNeedsResponse . |
| 4. The Test System responds with a SetChargingProfileResponse With status Accepted | 3. The CSMS sends a SetChargingProfileRequest <u>Note(s):</u> - If NotifyEVChargingNeedsResponseStatus was Processing, the Test System will wait 60 seconds for the request |
| 5. The Test System sends a NotifyEVChargingScheduleRequest . With evseld <Configured evseld> chargingSchedule <ChargingSchedule provided at step 3> | 6. The CSMS responds with a NotifyEVChargingScheduleResponse . |

Tool validations

* Step 2:

(Message: **NotifyEVChargingNeedsResponse**)**status** Accepted or Processing

* Step 3:

(Message: **SetChargingProfileRequest**)**evseld** <Configured evseld>**chargingProfilePurpose** TxProfile**transactionId** <Provided transactionId from before>

* Step 6:

(Message: **NotifyEVChargingScheduleResponse**)**status** Accepted

Post scenario validations:

N/a

TC_K_58_CSMS: Renegotiating a Charging Schedule - Initiated by CSMS

| | |
|-------------------|--|
| Test case name | Renegotiating a Charging Schedule - Initiated by CSMS |
| Test case Id | TC_K_58_CSMS |
| Use case Id(s) | K16 |
| Requirement(s) | K16.FR.06 |
| System under test | CSMS |
| Description | The CSMS sends a SetChargingProfileRequest to the Charging Station to influence the power or current drawn by the EV. The CSMS calculates a ChargingSchedule to stay within limits which MAY be imposed by an external system. |
| Purpose | To verify if the CSMS is able to renegotiate power/current drawn by the EV. |
| Prerequisite(s) | N/a |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State(s):

State is

[Authorized](#) AND[EVConnectedPreSession](#) AND[ISO15118SmartCharging](#)

Main (Test scenario)

| Charging Station | CSMS |
|--|---|
| 2. The Test System responds with a SetChargingProfileResponse With status <i>Accepted</i> | 1. The CSMS sends a SetChargingProfileRequest |
| 3. The Test System sends a NotifyEVChargingScheduleRequest . With evseld <i><Configured evseld></i> chargingSchedule <i><ChargingSchedule provided at step 1></i> | 4. The CSMS responds with a NotifyEVChargingScheduleResponse . |

Tool validations

* Step 1:

(Message: **SetChargingProfileRequest**)**evseld** *<Configured evseld>***chargingProfilePurpose** *TxProfile***transactionId** *<Provided transactionId from before>*

* Step 4:

(Message: **NotifyEVChargingScheduleResponse**)**status** *Accepted*

Post scenario validations:

N/a

TC_K_59_CSMS: Renegotiating a Charging Schedule - Initiated by CSMS - Send NotifyEVChargingNeeds

| | |
|-------------------|--|
| Test case name | Renegotiating a Charging Schedule - Initiated by CSMS - Send NotifyEVChargingNeeds |
| Test case Id | TC_K_59_CSMS |
| Use case Id(s) | K16 |
| Requirement(s) | K16.FR.12 |
| System under test | CSMS |
| Description | The CSMS sends a SetChargingProfileRequest to the Charging Station to influence the power or current drawn by the EV. The CSMS calculates a ChargingSchedule to stay within limits which MAY be imposed by an external system. |
| Purpose | To verify if the CSMS is able to handle a Charging Stations resending the charging needs of the EV. |
| Prerequisite(s) | N/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>Authorized</i> AND <i>EVConnectedPreSession</i> AND <i>ISO15118SmartCharging</i> |

| Main (Test scenario) | |
|--|---|
| Charging Station | CSMS |
| <u>Manual Action:</u> Trigger SetChargingProfile evseld <Configured evseld> chargingProfilePurpose TxProfile transactionId <Provided transactionId from before> | |
| 2. The Test System responds with a SetChargingProfileResponse With status <i>Accepted</i> | 1. The CSMS sends a SetChargingProfileRequest |
| 3. The Test System sends a NotifyEVChargingNeedsRequest . With evseld <Configured evseld> maxScheduleTuples & chargingNeeds <Configured values from mock EV> | 4. The CSMS responds with a NotifyEVChargingNeedsResponse . |
| 6. The Test System responds with a SetChargingProfileResponse With status <i>Accepted</i> | 5. The CSMS sends a SetChargingProfileRequest <u>Note(s)</u> : - If NotifyEVChargingNeedsResponseStatus was Processing, the Test System will wait 60 seconds for the request |
| 7. The Test System sends a NotifyEVChargingScheduleRequest . With evseld <Configured evseld> chargingSchedule <ChargingSchedule provided at step 3> | 8. The CSMS responds with a NotifyEVChargingScheduleResponse . |

| Tool validations |
|--|
| <div>* Step 1: (Message: SetChargingProfileRequest) evseld <Configured evseld> chargingProfilePurpose TxProfile transactionId <Provided transactionId from before></div> <div>* Step 4: (Message: NotifyEVChargingNeedsResponse) status Accepted or Processing</div> <div>* Step 5: (Message: SetChargingProfileRequest) evseld <Configured evseld> chargingProfilePurpose TxProfile transactionId <Provided transactionId from before></div> <div>* Step 8: (Message: NotifyEVChargingScheduleResponse) status Accepted</div> |
| <div>Post scenario validations: N/a</div> |

TC_K_70_CSMS: Set Charging Profile - Multiple Profiles

| | |
|-------------------|--|
| Test case name | Set Charging Profile - Multiple Profiles |
| Test case Id | TC_K_70_CSMS |
| Use case Id(s) | n/a |
| Requirement(s) | n/a |
| System under test | CSMS |
| Description | To enable the CSMS to influence the charging power or current drawn from a specific EVSE or the entire Charging Station over a period of time. The CSMS sends a SetChargingProfileRequest to the Charging Station to influence the power or current drawn by EVs. The CSMS calculates a ChargingSchedule to stay within certain limits, which MAY be imposed by any external system. |
| Purpose | To verify if the CSMS is able to set multiple Charging Profiles. |
| Prerequisite(s) | n/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: State is <i>EnergyTransferStarted</i> |

| Main (Test scenario) | |
|---|--|
| Charging Station | CSMS |
| 2. The Test System responds with a SetChargingProfileResponse with status <i>Accepted</i> | 1. The CSMS sends a SetChargingProfileRequest with chargingProfilePurpose <i>TxDefaultProfile</i> |
| 4. The Test System responds with a SetChargingProfileResponse with status <i>Accepted</i> | 3. The CSMS sends a SetChargingProfileRequest with chargingProfilePurpose <i>ChargingStationMaxProfile</i> |

| |
|---|
| Tool validations |
| <p>* Step 1: Message SetChargingProfileRequest chargingProfile.chargingProfilePurpose <i>TxDefaultProfile</i> chargingProfile.chargingProfileKind is <i>Absolute</i> OR <i>Recurring</i> chargingProfile.chargingSchedule.startSchedule must NOT be omitted. If chargingProfile.chargingProfileKind is <i>Recurring</i> then chargingProfile.recurrencyKind must NOT be omitted.</p> <p>* Step 3: Message SetChargingProfileRequest chargingProfile.id <different id from chargingProfile at step 1> chargingProfile.chargingProfilePurpose <i>ChargingStationMaxProfile</i> chargingProfile.chargingProfileKind is <i>Absolute</i> chargingProfile.chargingSchedule.startSchedule must NOT be omitted.</p> <p>The received Charging Profiles must comply with the requirements defined at part 2 specification.</p> |
| Post scenario validations: - N/a |

L Firmware Management

TC_L_01_CSMS: Secure Firmware Update - Installation successful

| | |
|-------------------|---|
| Test case name | Secure Firmware Update - Installation successful |
| Test case Id | TC_L_01_CSMS |
| Use case Id(s) | L01 |
| Requirement(s) | L01.FR.01,L01.FR.11,L01.FR.15 |
| System under test | CSMS |
| Description | The CSMS is able to request the Charging Station to securely download and install a new firmware by sending an UpdateFirmwareRequest with a signingCertificate. |
| Purpose | To verify if the CSMS is able to request the Charging Station to securely download and install a new firmware. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a UpdateFirmwareResponse With status <i>Accepted</i> | 1. The CSMS sends a UpdateFirmwareRequest |
| 3. The Test System sends a FirmwareStatusNotificationRequest . With status <i>Downloading</i> | 4. The CSMS responds with a FirmwareStatusNotificationResponse . |
| 5. The Test System sends a FirmwareStatusNotificationRequest . With status <i>Downloaded</i> | 6. The CSMS responds with a FirmwareStatusNotificationResponse . |
| 7. The Test System sends a FirmwareStatusNotificationRequest . With status <i>SignatureVerified</i> | 8. The CSMS responds with a FirmwareStatusNotificationResponse . |
| 9. The Test System sends a FirmwareStatusNotificationRequest . With status <i>Installing</i> | 10. The CSMS responds with a FirmwareStatusNotificationResponse . |
| 11. The Test System sends a FirmwareStatusNotificationRequest . With status <i>InstallRebooting</i> | 12. The CSMS responds with a FirmwareStatusNotificationResponse . |
| 13. The Test System sends a BootNotificationRequest With reason <i>FirmwareUpdate</i> | 14. The CSMS responds with a BootNotificationResponse |

| Main (Test scenario) | |
|---|---|
| <p>15. The Test System notifies the CSMS about the current state of all connectors.</p> <p>Message: StatusNotificationRequest connectorStatus <i>Available</i> Message: NotifyEventRequest trigger <i>Delta</i> actualValue <i>"Available"</i> component.name <i>"Connector"</i> variable.name <i>"AvailabilityState"</i></p> | <p>16. The CSMS responds accordingly.</p> |
| <p>17. The Test System sends a FirmwareStatusNotificationRequest. With status <i>Installed</i></p> | <p>18. The CSMS responds with a FirmwareStatusNotificationResponse.</p> |

| Tool validations |
|---|
| <p>* Step 1: Message UpdateFirmwareRequest - firmware.signingCertificate <i><Configured signingCertificate></i> - firmware.signature <i><Configured signature></i></p> <p>* Step 14: Message BootNotificationResponse - status <i>Accepted</i></p> |
| <p>Post scenario validations: N/a</p> |

TC_L_02_CSMS: Secure Firmware Update - InstallScheduled

| | |
|--------------------------|--|
| Test case name | Secure Firmware Update - InstallScheduled |
| Test case Id | TC_L_02_CSMS |
| Use case Id(s) | L01 |
| Requirement(s) | L01.FR.01,L01.FR.11,L01.FR.15 |
| System under test | CSMS |
| Description | The CSMS is able to request the Charging Station to securely download and install a new firmware by sending an UpdateFirmwareRequest with a signingCertificate . |
| Purpose | To verify if the CSMS is able to request the Charging Station to securely download a new firmware and install it |
| Prerequisite(s) | The CSMS configuration firmware installDateTime needs to be set to a future dateTime . |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a UpdateFirmwareResponse With status <i>Accepted</i> | 1. The CSMS sends a UpdateFirmwareRequest |
| 3. The Test System sends a FirmwareStatusNotificationRequest . With status <i>Downloading</i> | 4. The CSMS responds with a FirmwareStatusNotificationResponse . |
| 5. The Test System sends a FirmwareStatusNotificationRequest . With status <i>Downloaded</i> | 6. The CSMS responds with a FirmwareStatusNotificationResponse . |
| 7. The Test System sends a FirmwareStatusNotificationRequest . With status <i>SignatureVerified</i> | 8. The CSMS responds with a FirmwareStatusNotificationResponse . |
| 9. The Test System sends a FirmwareStatusNotificationRequest . With status <i>InstallScheduled</i> | 10. The CSMS responds with a FirmwareStatusNotificationResponse . |
| 11. The Test System sends a FirmwareStatusNotificationRequest . With status <i>Installing</i> <u>Note(s):</u> - This step will be executed after the given installDateTime from step 1 has been reached. | 12. The CSMS responds with a FirmwareStatusNotificationResponse . |
| 13. The Test System sends a FirmwareStatusNotificationRequest . With status <i>InstallRebooting</i> | 14. The CSMS responds with a FirmwareStatusNotificationResponse . |
| 15. The Test System sends a BootNotificationRequest With reason <i>FirmwareUpdate</i> | 16. The CSMS responds with a BootNotificationResponse |

| Main (Test scenario) | |
|--|--|
| <p>17. The Test System notifies the CSMS about the current state of all connectors.</p> <p>Message: StatusNotificationRequest connectorStatus <i>Available</i> Message: NotifyEventRequest trigger <i>Delta</i> actualValue <i>"Available"</i> component.name <i>"Connector"</i> variable.name <i>"AvailabilityState"</i></p> | <p>18. The CSMS responds accordingly.</p> |
| <p>19. The Test System sends a FirmwareStatusNotificationRequest. With status <i>Installed</i></p> | <p>20. The CSMS responds with a FirmwareStatusNotificationResponse.</p> |

| Tool validations |
|---|
| <p>* Step 1: Message UpdateFirmwareRequest - firmware.installDateTime <i><A dateTime in the future></i></p> <p>* Step 16: Message BootNotificationResponse - status <i>Accepted</i></p> |
| <p>Post scenario validations: N/a</p> |

TC_L_03_CSMS: Secure Firmware Update - DownloadScheduled

| | |
|--------------------------|--|
| Test case name | Secure Firmware Update - DownloadScheduled |
| Test case Id | TC_L_03_CSMS |
| Use case Id(s) | L01 |
| Requirement(s) | L01.FR.01,L01.FR.11,L01.FR.15 |
| System under test | CSMS |
| Description | The CSMS is able to request the Charging Station to securely download and install a new firmware by sending an UpdateFirmwareRequest with a signingCertificate . |
| Purpose | To verify if the CSMS is able to request the Charging Station to schedule securely downloading a new firmware. |
| Prerequisite(s) | The CSMS configuration firmware retrieveDateTime needs to be set to a future dateTime . |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a UpdateFirmwareResponse With status <i>Accepted</i> | 1. The CSMS sends a UpdateFirmwareRequest |
| 3. The Test System sends a FirmwareStatusNotificationRequest . With status <i>DownloadScheduled</i> | 4. The CSMS responds with a FirmwareStatusNotificationResponse . |
| 5. The Test System sends a FirmwareStatusNotificationRequest . With status <i>Downloading</i> <u>Note(s):</u> - This step will be executed after the given retrieveDateTime from step 1 has been reached. | 6. The CSMS responds with a FirmwareStatusNotificationResponse . |
| 7. The Test System sends a FirmwareStatusNotificationRequest . With status <i>Downloaded</i> | 8. The CSMS responds with a FirmwareStatusNotificationResponse . |
| 9. The Test System sends a FirmwareStatusNotificationRequest . With status <i>SignatureVerified</i> | 10. The CSMS responds with a FirmwareStatusNotificationResponse . |
| 11. The Test System sends a FirmwareStatusNotificationRequest . With status <i>Installing</i> | 12. The CSMS responds with a FirmwareStatusNotificationResponse . |
| 13. The Test System sends a FirmwareStatusNotificationRequest . With status <i>InstallRebooting</i> | 14. The CSMS responds with a FirmwareStatusNotificationResponse . |
| 15. The Test System sends a BootNotificationRequest With reason <i>FirmwareUpdate</i> | 16. The CSMS responds with a BootNotificationResponse |

| Main (Test scenario) | |
|---|---|
| <p>17. The Test System notifies the CSMS about the current state of all connectors.</p> <p>Message: StatusNotificationRequest connectorStatus <i>Available</i> Message: NotifyEventRequest trigger <i>Delta</i> actualValue <i>"Available"</i> component.name <i>"Connector"</i> variable.name <i>"AvailabilityState"</i></p> | <p>18. The CSMS responds accordingly.</p> |
| <p>19. The Test System sends a FirmwareStatusNotificationRequest. With status <i>Installed</i></p> | <p>20. The CSMS responds with a FirmwareStatusNotificationResponse.</p> |

| Tool validations |
|--|
| <p>* Step 1: Message UpdateFirmwareRequest - firmware.retrieveDateTime <i><A dateTime in the future></i></p> <p>* Step 16: Message BootNotificationResponse - status <i>Accepted</i></p> |
| <p>Post scenario validations: N/a</p> |

TC_L_04_CSMS: Secure Firmware Update - RevokedCertificate

| | |
|-------------------|--|
| Test case name | Secure Firmware Update - RevokedCertificate |
| Test case Id | TC_L_04_CSMS |
| Use case Id(s) | L01 |
| Requirement(s) | L01.FR.01 |
| System under test | CSMS |
| Description | The CSMS is able to request the Charging Station to securely download and install a new firmware by sending an UpdateFirmwareRequest with a signingCertificate . |
| Purpose | To verify if the CSMS is able to handle a Charging Station reporting the firmware signing certificate is revoked. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a UpdateFirmwareResponse With status <i>RevokedCertificate</i> | 1. The CSMS sends a UpdateFirmwareRequest |

| |
|-----------------------------------|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_L_05_CSMS: Secure Firmware Update - InvalidCertificate

| | |
|-------------------|---|
| Test case name | Secure Firmware Update - InvalidCertificate |
| Test case Id | TC_L_05_CSMS |
| Use case Id(s) | L01 |
| Requirement(s) | L01.FR.01 |
| System under test | CSMS |
| Description | The CSMS is able to request the Charging Station to securely download and install a new firmware by sending an UpdateFirmwareRequest with a signingCertificate. |
| Purpose | To verify if the CSMS is able to handle a Charging Station reporting the firmware signing certificate is invalid. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a UpdateFirmwareResponse With status <i>InvalidCertificate</i> | 1. The CSMS sends a UpdateFirmwareRequest |
| 3. The Test System sends a SecurityEventNotificationRequest With type is <i>InvalidFirmwareSigningCertificate</i> | 4. The CSMS responds with a SecurityEventNotificationResponse |

| |
|-----------------------------------|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_L_06_CSMS: Secure Firmware Update - InvalidSignature

| | |
|--------------------------|---|
| Test case name | Secure Firmware Update - InvalidSignature |
| Test case Id | TC_L_06_CSMS |
| Use case Id(s) | L01 |
| Requirement(s) | L01.FR.01,L01.FR.11 |
| System under test | CSMS |
| Description | The CSMS is able to request the Charging Station to securely download and install a new firmware by sending an UpdateFirmwareRequest with a signingCertificate. |
| Purpose | To verify if the CSMS is able to handle a Charging Station reporting the signature is invalid. |
| Prerequisite(s) | N/a |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a UpdateFirmwareResponse With status <i>Accepted</i> | 1. The CSMS sends a UpdateFirmwareRequest |
| 3. The Test System sends a FirmwareStatusNotificationRequest . With status <i>Downloading</i> | 4. The CSMS responds with a FirmwareStatusNotificationResponse . |
| 5. The Test System sends a FirmwareStatusNotificationRequest . With status <i>Downloaded</i> | 6. The CSMS responds with a FirmwareStatusNotificationResponse . |
| 7. The Test System sends a FirmwareStatusNotificationRequest . With status <i>InvalidSignature</i> | 8. The CSMS responds with a FirmwareStatusNotificationResponse . |
| 9. The Test System sends a SecurityEventNotificationRequest With type is <i>InvalidFirmwareSignature</i> | 10. The CSMS responds with a SecurityEventNotificationResponse |

| |
|--|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_L_07_CSMS: Secure Firmware Update - DownloadFailed

| | |
|--------------------------|--|
| Test case name | Secure Firmware Update - DownloadFailed |
| Test case Id | TC_L_07_CSMS |
| Use case Id(s) | L01 |
| Requirement(s) | L01.FR.01,L01.FR.11 |
| System under test | CSMS |
| Description | The CSMS is able to request the Charging Station to securely download and install a new firmware by sending an <code>UpdateFirmwareRequest</code> with a <code>signingCertificate</code> . |
| Purpose | To verify if the CSMS is able to handle a Charging Station reporting it failed to download the firmware. |
| Prerequisite(s) | N/a |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a UpdateFirmwareResponse With status <i>Accepted</i> | 1. The CSMS sends a UpdateFirmwareRequest |
| 3. The Test System sends a FirmwareStatusNotificationRequest . With status <i>Downloading</i> | 4. The CSMS responds with a FirmwareStatusNotificationResponse . |
| 5. The Test System sends a FirmwareStatusNotificationRequest . With status <i>DownloadFailed</i> | 6. The CSMS responds with a FirmwareStatusNotificationResponse . |

| |
|--|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_L_08_CSMS: Secure Firmware Update - InstallVerificationFailed

| | |
|-------------------|---|
| Test case name | Secure Firmware Update - InstallVerificationFailed |
| Test case Id | TC_L_08_CSMS |
| Use case Id(s) | L01 |
| Requirement(s) | L01.FR.01,L01.FR.11 |
| System under test | CSMS |
| Description | The CSMS is able to request the Charging Station to securely download and install a new firmware by sending an UpdateFirmwareRequest with a signingCertificate. |
| Purpose | To verify if the CSMS is able to handle a Charging Station reporting the verification of the firmware failed during installation. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a UpdateFirmwareResponse With status <i>Accepted</i> | 1. The CSMS sends a UpdateFirmwareRequest |
| 3. The Test System sends a FirmwareStatusNotificationRequest . With status <i>Downloading</i> | 4. The CSMS responds with a FirmwareStatusNotificationResponse . |
| 5. The Test System sends a FirmwareStatusNotificationRequest . With status <i>Downloaded</i> | 6. The CSMS responds with a FirmwareStatusNotificationResponse . |
| 7. The Test System sends a FirmwareStatusNotificationRequest . With status <i>SignatureVerified</i> | 8. The CSMS responds with a FirmwareStatusNotificationResponse . |
| 9. The Test System sends a FirmwareStatusNotificationRequest . With status <i>Installing</i> | 10. The CSMS responds with a FirmwareStatusNotificationResponse . |
| 11. The Test System sends a FirmwareStatusNotificationRequest . With status <i>InstallVerificationFailed</i> | 12. The CSMS responds with a FirmwareStatusNotificationResponse . |

| |
|-----------------------------------|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_L_09_CSMS: Secure Firmware Update - InstallationFailed

| | |
|-------------------|--|
| Test case name | Secure Firmware Update - InstallationFailed |
| Test case Id | TC_L_09_CSMS |
| Use case Id(s) | L01 |
| Requirement(s) | L01.FR.01,L01.FR.11 |
| System under test | CSMS |
| Description | The CSMS is able to request the Charging Station to securely download and install a new firmware by sending an UpdateFirmwareRequest with a signingCertificate . |
| Purpose | To verify if the CSMS is able to handle a Charging Station reporting the installation of the firmware failed. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a UpdateFirmwareResponse With status <i>Accepted</i> | 1. The CSMS sends a UpdateFirmwareRequest |
| 3. The Test System sends a FirmwareStatusNotificationRequest . With status <i>Downloading</i> | 4. The CSMS responds with a FirmwareStatusNotificationResponse . |
| 5. The Test System sends a FirmwareStatusNotificationRequest . With status <i>Downloaded</i> | 6. The CSMS responds with a FirmwareStatusNotificationResponse . |
| 7. The Test System sends a FirmwareStatusNotificationRequest . With status <i>SignatureVerified</i> | 8. The CSMS responds with a FirmwareStatusNotificationResponse . |
| 9. The Test System sends a FirmwareStatusNotificationRequest . With status <i>Installing</i> | 10. The CSMS responds with a FirmwareStatusNotificationResponse . |
| 11. The Test System sends a FirmwareStatusNotificationRequest . With status <i>InstallRebooting</i> | 12. The CSMS responds with a FirmwareStatusNotificationResponse . |
| 13. The Test System sends a BootNotificationRequest With reason <i>FirmwareUpdate</i> | 14. The CSMS responds with a BootNotificationResponse |
| 15. The Test System notifies the CSMS about the current state of all connectors. Message: StatusNotificationRequest connectorStatus <i>Available</i> Message: NotifyEventRequest trigger <i>Delta</i> actualValue <i>"Available"</i> component.name <i>"Connector"</i> variable.name <i>"AvailabilityState"</i> | 16. The CSMS responds accordingly. |

| Main (Test scenario) | |
|--|--|
| 17. The Test System sends a FirmwareStatusNotificationRequest . With status <i>InstallationFailed</i> | 18. The CSMS responds with a FirmwareStatusNotificationResponse . |

| Tool validations |
|--|
| * Step 14: Message BootNotificationResponse - status <i>Accepted</i> |
| Post scenario validations: N/a |

TC_L_10_CSMS: Secure Firmware Update - AcceptedCanceled

| | |
|--------------------------|--|
| Test case name | Secure Firmware Update - AcceptedCanceled |
| Test case Id | TC_L_10_CSMS |
| Use case Id(s) | L01 |
| Requirement(s) | L01.FR.01,L01.FR.11,L01.FR.24 |
| System under test | CSMS |
| Description | The CSMS is able to request the Charging Station to securely download and install a new firmware by sending an UpdateFirmwareRequest with a signingCertificate . |
| Purpose | To verify if the CSMS is able to handle a Charging Station reporting an ongoing installation of a firmware was canceled and it is now starting the new firmware update. |
| Prerequisite(s) | The CSMS is able to request a new firmware update, while there is already one ongoing on the Charging Station. |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a UpdateFirmwareResponse With status Accepted | 1. The CSMS sends a UpdateFirmwareRequest |
| 3. The Test System sends a FirmwareStatusNotificationRequest . With status Downloading | 4. The CSMS responds with a FirmwareStatusNotificationResponse . |
| 6. The Test System responds with a UpdateFirmwareResponse With status AcceptedCanceled | 5. The CSMS sends a UpdateFirmwareRequest |
| 7. The Test System sends a FirmwareStatusNotificationRequest . With status Downloading | 8. The CSMS responds with a FirmwareStatusNotificationResponse . |
| 9. The Test System sends a FirmwareStatusNotificationRequest . With status Downloaded | 10. The CSMS responds with a FirmwareStatusNotificationResponse . |
| 11. The Test System sends a FirmwareStatusNotificationRequest . With status SignatureVerified | 12. The CSMS responds with a FirmwareStatusNotificationResponse . |
| 13. The Test System sends a FirmwareStatusNotificationRequest . With status Installing | 14. The CSMS responds with a FirmwareStatusNotificationResponse . |
| 15. The Test System sends a FirmwareStatusNotificationRequest . With status InstallRebooting | 16. The CSMS responds with a FirmwareStatusNotificationResponse . |
| 17. The Test System sends a BootNotificationRequest With reason FirmwareUpdate | 18. The CSMS responds with a BootNotificationResponse |

| Main (Test scenario) | |
|---|---|
| <p>19. The Test System notifies the CSMS about the current state of all connectors.</p> <p>Message: StatusNotificationRequest connectorStatus <i>Available</i> Message: NotifyEventRequest trigger <i>Delta</i> actualValue <i>"Available"</i> component.name <i>"Connector"</i> variable.name <i>"AvailabilityState"</i></p> | <p>20. The CSMS responds accordingly.</p> |
| <p>21. The Test System sends a FirmwareStatusNotificationRequest. With status <i>Installed</i></p> | <p>22. The CSMS responds with a FirmwareStatusNotificationResponse.</p> |

| Tool validations |
|---|
| <p>* Step 18: Message BootNotificationResponse - status <i>Accepted</i></p> |
| <p>Post scenario validations: N/a</p> |

TC_L_11_CSMS: Secure Firmware Update - Unable to cancel

| | |
|--------------------------|--|
| Test case name | Secure Firmware Update - Unable to cancel |
| Test case Id | TC_L_11_CSMS |
| Use case Id(s) | L01 |
| Requirement(s) | L01.FR.01,L01.FR.11,L01.FR.27 |
| System under test | CSMS |
| Description | The CSMS is able to request the Charging Station to securely download and install a new firmware by sending an UpdateFirmwareRequest with a signingCertificate . |
| Purpose | To verify if the CSMS is able to handle a Charging Station reporting the ongoing installation of a firmware cannot be canceled. |
| Prerequisite(s) | The CSMS is able to request a new firmware update, while there is already one ongoing on the Charging Station. |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a UpdateFirmwareResponse With status <i>Accepted</i> | 1. The CSMS sends a UpdateFirmwareRequest |
| 3. The Test System sends a FirmwareStatusNotificationRequest . With status <i>Downloading</i> | 4. The CSMS responds with a FirmwareStatusNotificationResponse . |
| 6. The Test System responds with a UpdateFirmwareResponse With status <i>Rejected</i> | 5. The CSMS sends a UpdateFirmwareRequest |
| 7. The Test System sends a FirmwareStatusNotificationRequest . With status <i>Downloaded</i> | 8. The CSMS responds with a FirmwareStatusNotificationResponse . |
| 9. The Test System sends a FirmwareStatusNotificationRequest . With status <i>SignatureVerified</i> | 10. The CSMS responds with a FirmwareStatusNotificationResponse . |
| 11. The Test System sends a FirmwareStatusNotificationRequest . With status <i>Installing</i> | 12. The CSMS responds with a FirmwareStatusNotificationResponse . |
| 13. The Test System sends a FirmwareStatusNotificationRequest . With status <i>InstallRebooting</i> | 14. The CSMS responds with a FirmwareStatusNotificationResponse . |
| 15. The Test System sends a BootNotificationRequest With reason <i>FirmwareUpdate</i> | 16. The CSMS responds with a BootNotificationResponse |

| Main (Test scenario) | |
|---|---|
| <p>17. The Test System notifies the CSMS about the current state of all connectors.</p> <p>Message: StatusNotificationRequest connectorStatus <i>Available</i> Message: NotifyEventRequest trigger <i>Delta</i> actualValue <i>"Available"</i> component.name <i>"Connector"</i> variable.name <i>"AvailabilityState"</i></p> | <p>18. The CSMS responds accordingly.</p> |
| <p>19. The Test System sends a FirmwareStatusNotificationRequest. With status <i>Installed</i></p> | <p>20. The CSMS responds with a FirmwareStatusNotificationResponse.</p> |

| Tool validations |
|---|
| <p>* Step 16: Message BootNotificationResponse - status <i>Accepted</i></p> |
| <p>Post scenario validations: N/a</p> |

TC_L_13_CSMS: Secure Firmware Update - Unable to download/install firmware with ongoing transaction - AllowNewSessionsPendingFirmwareUpdate is false

| | |
|-------------------|---|
| Test case name | Secure Firmware Update - Unable to download/install firmware with ongoing transaction - AllowNewSessionsPendingFirmwareUpdate is false |
| Test case Id | TC_L_13_CSMS |
| Use case Id(s) | L01 |
| Requirement(s) | L01.FR.01,L01.FR.11 |
| System under test | CSMS |
| Description | The CSMS is able to request the Charging Station to securely download and install a new firmware by sending an UpdateFirmwareRequest with a signingCertificate. |
| Purpose | To verify if the CSMS is able to handle a Charging Station setting connectors to Unavailable while preparing a firmware update when there is a transaction ongoing. |
| Prerequisite(s) | The CSMS is able to request a new firmware update when there is a transaction ongoing on the Charging Station. |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): State is <i>EnergyTransferStarted</i> |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a UpdateFirmwareResponse With status <i>Accepted</i> | 1. The CSMS sends a UpdateFirmwareRequest |
| 3. The Test System sends a FirmwareStatusNotificationRequest . With status <i>DownloadScheduled</i> | 4. The CSMS responds with a FirmwareStatusNotificationResponse . |
| 5. The Test System notifies the CSMS about the state change of all connectors that don't have a running transaction. Message: StatusNotificationRequest connectorStatus <i>Unavailable</i> Message: NotifyEventRequest trigger <i>Delta</i> actualValue <i>"Unavailable"</i> component.name <i>"Connector"</i> variable.name <i>"AvailabilityState"</i> | 6. The CSMS responds accordingly. |
| 7. Execute Reusable State <i>StopAuthorized</i> <u>Note(s)</u> Wait <configured transaction duration> before executing this step | |
| 8. Execute Reusable State <i>EVConnectedPostSession</i> | |
| 9. Execute Reusable State <i>EVDisconnected</i> | |
| 10. The Test System sends a FirmwareStatusNotificationRequest . With status <i>Downloading</i> <u>Note(s)</u> : - This step will be executed after the given retrieveDateTime from step 1 has been reached. | 11. The CSMS responds with a FirmwareStatusNotificationResponse . |

| Main (Test scenario) | |
|--|---|
| 12. The Test System sends a FirmwareStatusNotificationRequest . With status <i>Downloaded</i> | 13. The CSMS responds with a FirmwareStatusNotificationResponse . |
| 14. The Test System sends a FirmwareStatusNotificationRequest . With status <i>SignatureVerified</i> | 15. The CSMS responds with a FirmwareStatusNotificationResponse . |
| 16. The Test System sends a FirmwareStatusNotificationRequest . With status <i>Installing</i> | 17. The CSMS responds with a FirmwareStatusNotificationResponse . |
| 18. The Test System sends a FirmwareStatusNotificationRequest . With status <i>InstallRebooting</i> | 19. The CSMS responds with a FirmwareStatusNotificationResponse . |
| 20. The Test System sends a BootNotificationRequest With reason <i>FirmwareUpdate</i> | 21. The CSMS responds with a BootNotificationResponse |
| 22. The Test System notifies the CSMS about the current state of all connectors. Message: StatusNotificationRequest connectorStatus <i>Available</i> Message: NotifyEventRequest trigger <i>Delta</i> actualValue <i>"Available"</i> component.name <i>"Connector"</i> variable.name <i>"AvailabilityState"</i> | 23. The CSMS responds accordingly. |
| 24. The Test System sends a FirmwareStatusNotificationRequest . With status <i>Installed</i> | 25. The CSMS responds with a FirmwareStatusNotificationResponse . |

| Tool validations |
|--|
| * Step 1: Message UpdateFirmwareRequest - firmware.signingCertificate <i><configured signingCertificate></i> * Step 19: Message BootNotificationResponse - status <i>Accepted</i> |
| Post scenario validations: N/a |

TC_L_17_CSMS: Publish Firmware - Published

| | |
|--------------------------|--|
| Test case name | Publish Firmware - Published |
| Test case Id | TC_L_17_CSMS |
| Use case Id(s) | L03 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | The Local Controller downloads and publishes a firmware update at the specified URL. This allows the CSMS to send UpdateFirmwareRequests with the URI pointing to the Local Controller, to any Charging Station connected to the Local Controller. This allows the site to save bandwidth and data on the WAN interface. |
| Purpose | To verify if the CSMS is able to publish a firmware on the local controller as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a PublishFirmwareResponse with status <i>Accepted</i> | 1. The CSMS sends a PublishFirmwareRequest |
| 3. The Test System sends a PublishFirmwareStatusNotificationRequest with status <i>Downloading</i> | 4. The CSMS responds with a PublishFirmwareStatusNotificationResponse |
| 5. The Test System sends a PublishFirmwareStatusNotificationRequest with status <i>Downloaded</i> | 6. The CSMS responds with a PublishFirmwareStatusNotificationResponse |
| 7. The Test System sends a PublishFirmwareStatusNotificationRequest with status <i>ChecksumVerified</i> | 8. The CSMS responds with a PublishFirmwareStatusNotificationResponse |
| 9. The Test System sends a PublishFirmwareStatusNotificationRequest with status <i>Published</i> AND location <i><Configured firmware_location></i> | 10. The CSMS responds with a PublishFirmwareStatusNotificationResponse |

| |
|---|
| Tool validations |
| * Step 1: Message PublishFirmwareRequest - location <i><Configured firmware_location></i> |
| Post scenario validations: - N/a |

TC_L_24_CSMS: Publish Firmware - Download failed

| | |
|--------------------------|--|
| Test case name | Publish Firmware - Download failed |
| Test case Id | TC_L_24_CSMS |
| Use case Id(s) | L03 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | The Local Controller downloads and publishes a firmware update at the specified URL. This allows the CSMS to send UpdateFirmwareRequests with the URI pointing to the Local Controller, to any Charging Station connected to the Local Controller. This allows the site to save bandwidth and data on the WAN interface. |
| Purpose | To verify if the CSMS is able to publish a firmware on the local controller as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a PublishFirmwareResponse with status <i>Accepted</i> | 1. The CSMS sends a PublishFirmwareRequest |
| 3. The Test System sends a PublishFirmwareStatusNotificationRequest with status <i>Downloading</i> | 4. The CSMS responds with a PublishFirmwareStatusNotificationResponse |
| 5. The Test System sends a PublishFirmwareStatusNotificationRequest with status <i>DownloadFailed</i> | 6. The CSMS responds with a PublishFirmwareStatusNotificationResponse |

| |
|--|
| Tool validations |
| * Step 1: Message PublishFirmwareRequest - location <Configured firmware_location> |
| Post scenario validations: - N/a |

TC_L_19_CSMS: Publish Firmware - Invalid Checksum

| | |
|--------------------------|--|
| Test case name | Publish Firmware - Invalid Checksum |
| Test case Id | TC_L_19_CSMS |
| Use case Id(s) | L03 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | The Local Controller downloads and publishes a firmware update at the specified URL. This allows the CSMS to send UpdateFirmwareRequests with the URI pointing to the Local Controller, to any Charging Station connected to the Local Controller. This allows the site to save bandwidth and data on the WAN interface. |
| Purpose | To verify if the CSMS is able to publish a firmware on the local controller as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a PublishFirmwareResponse with status <i>Accepted</i> | 1. The CSMS sends a PublishFirmwareRequest |
| 3. The Test System sends a PublishFirmwareStatusNotificationRequest with status <i>Downloading</i> | 4. The CSMS responds with a PublishFirmwareStatusNotificationResponse |
| 5. The Test System sends a PublishFirmwareStatusNotificationRequest with status <i>Downloaded</i> | 6. The CSMS responds with a PublishFirmwareStatusNotificationResponse |
| 7. The Test System sends a PublishFirmwareStatusNotificationRequest with status <i>InvalidChecksum</i> | 8. The CSMS responds with a PublishFirmwareStatusNotificationResponse |

| |
|--|
| Tool validations |
| * Step 1: Message PublishFirmwareRequest - location <Configured firmware_location> |
| Post scenario validations: - N/a |

TC_L_20_CSMS: Publish Firmware - PublishFailed

| | |
|--------------------------|--|
| Test case name | Publish Firmware - PublishFailed |
| Test case Id | TC_L_20_CSMS |
| Use case Id(s) | L03 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | The Local Controller downloads and publishes a firmware update at the specified URL. This allows the CSMS to send UpdateFirmwareRequests with the URI pointing to the Local Controller, to any Charging Station connected to the Local Controller. This allows the site to save bandwidth and data on the WAN interface. |
| Purpose | To verify if the CSMS is able to publish a firmware on the local controller as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a PublishFirmwareResponse with status <i>Accepted</i> | 1. The CSMS sends a PublishFirmwareRequest |
| 3. The Test System sends a PublishFirmwareStatusNotificationRequest with status <i>Downloading</i> | 4. The CSMS responds with a PublishFirmwareStatusNotificationResponse |
| 5. The Test System sends a PublishFirmwareStatusNotificationRequest with status <i>Downloaded</i> | 6. The CSMS responds with a PublishFirmwareStatusNotificationResponse |
| 7. The Test System sends a PublishFirmwareStatusNotificationRequest with status <i>ChecksumVerified</i> | 8. The CSMS responds with a PublishFirmwareStatusNotificationResponse |
| 9. The Test System sends a PublishFirmwareStatusNotificationRequest with status <i>PublishFailed</i> | 10. The CSMS responds with a PublishFirmwareStatusNotificationResponse |

| |
|--|
| Tool validations |
| * Step 1: Message PublishFirmwareRequest - location <Configured firmware_location> |
| Post scenario validations: - N/a |

TC_L_21_CSMS: Unpublish Firmware - Unpublished

| | |
|-------------------|---|
| Test case name | Unpublish Firmware - Unpublished |
| Test case Id | TC_L_21_CSMS |
| Use case Id(s) | L04 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | Stop serving a firmware update to connected Charging Stations. |
| Purpose | To verify if the CSMS is able to unpublish a firmware on the local controller as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a UnpublishFirmwareResponse with status <i>Unpublished</i> | 1. The CSMS sends a UnpublishFirmwareRequest |

| |
|-------------------------------------|
| Tool validations |
| -N/a |
| Post scenario validations: - N/a |

TC_L_22_CSMS: Unpublish Firmware - NoFirmware

| | |
|--------------------------|---|
| Test case name | Unpublish Firmware - NoFirmware |
| Test case Id | TC_L_22_CSMS |
| Use case Id(s) | L04 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | Stop serving a firmware update to connected Charging Stations. |
| Purpose | To verify if the CSMS is able to unpublish a firmware on the local controller as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a UnpublishFirmwareResponse with status NoFirmware | 1. The CSMS sends a UnpublishFirmwareRequest |

| |
|--|
| Tool validations |
| -N/a |
| Post scenario validations: - N/a |

TC_L_23_CSMS: Unpublish Firmware - Download Ongoing

| | |
|--------------------------|---|
| Test case name | Unpublish Firmware - Download Ongoing |
| Test case Id | TC_L_23_CSMS |
| Use case Id(s) | L04 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | Stop serving a firmware update to connected Charging Stations. |
| Purpose | To verify if the CSMS is able to unpublish a firmware on the local controller as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a UnpublishFirmwareResponse with status <i>DownloadOngoing</i> | 1. The CSMS sends a UnpublishFirmwareRequest |

| |
|--|
| Tool validations |
| -N/a |
| Post scenario validations: - N/a |

M Certificate Management

TC_M_01_CSMS: Install CA certificate - CSMSRootCertificate

| | |
|-------------------|---|
| Test case name | Install CA certificate - CSMSRootCertificate |
| Test case Id | TC_M_01_CSMS |
| Use case Id(s) | M05 |
| Requirement(s) | M05.FR.01 |
| System under test | CSMS |
| Description | The CSMS is able to request the Charging Station to install new Root CA certificates using the InstallCertificateRequest message. |
| Purpose | To verify if the CSMS is able to request a Charging Station to install a new CSMSRootCertificate. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>CertificateInstalled</i> for certificateType CSMSRootCertificate | |

| |
|-----------------------------------|
| Tool validations |
| N.a |
| Post scenario validations: N/a |

TC_M_02_CSMS: Install CA certificate - ManufacturerRootCertificate

| | |
|-------------------|---|
| Test case name | Install CA certificate - ManufacturerRootCertificate |
| Test case Id | TC_M_02_CSMS |
| Use case Id(s) | M05 |
| Requirement(s) | M05.FR.01 |
| System under test | CSMS |
| Description | The CSMS is able to request the Charging Station to install new Root CA certificates using the InstallCertificateRequest message. |
| Purpose | To verify if the CSMS is able to request a Charging Station to install a new ManufacturerRootCertificate. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>CertificateInstalled</i> for certificateType <i>ManufacturerRootCertificate</i> | |

| |
|-----------------------------------|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_M_03_CSMS: Install CA certificate - V2GRootCertificate

| | |
|--------------------------|---|
| Test case name | Install CA certificate - V2GRootCertificate |
| Test case Id | TC_M_03_CSMS |
| Use case Id(s) | M05 |
| Requirement(s) | M05.FR.01 |
| System under test | CSMS |
| Description | The CSMS is able to request the Charging Station to install new Root CA certificates using the InstallCertificateRequest message. |
| Purpose | To verify if the CSMS is able to request a Charging Station to install a new V2GRootCertificate. |
| Prerequisite(s) | N/a |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>CertificateInstalled</i> for certificateType V2GRootCertificate | |

| |
|--|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_M_04_CSMS: Install CA certificate - MORootCertificate

| | |
|-------------------|---|
| Test case name | Install CA certificate - MORootCertificate |
| Test case Id | TC_M_04_CSMS |
| Use case Id(s) | M05 |
| Requirement(s) | M05.FR.01 |
| System under test | CSMS |
| Description | The CSMS is able to request the Charging Station to install new Root CA certificates using the InstallCertificateRequest message. |
| Purpose | To verify if the CSMS is able to request a Charging Station to install a new MORootCertificate. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>CertificateInstalled</i> for certificateType <i>MORootCertificate</i> | |

| |
|-----------------------------------|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_M_05_CSMS: Install CA certificate - Failed

| | |
|-------------------|---|
| Test case name | Install CA certificate - Failed |
| Test case Id | TC_M_05_CSMS |
| Use case Id(s) | M05 |
| Requirement(s) | M05.FR.01,M05.FR.03 |
| System under test | CSMS |
| Description | The CSMS is able to request the Charging Station to install new Root CA certificates using the InstallCertificateRequest message. |
| Purpose | To verify if the CSMS is able to handle a Charging Station reporting it failed to install the requested certificate. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action:</u> Trigger the CSMS to send an InstallCertificateRequest with certificateType CSMSRootCertificate. | |
| 2. The Test System responds with a InstallCertificateResponse With status is Failed | 1. The CSMS sends a InstallCertificateRequest |

| |
|--|
| Tool validations |
| * Step 1: Message: InstallCertificateRequest certificateType must be CSMSRootCertificate certificate contains <A certificate> |
| Post scenario validations: N/a |

TC_M_12_CSMS: Retrieve certificates from Charging Station - CSMSRootCertificate

| | |
|--------------------------|---|
| Test case name | Retrieve certificates from Charging Station - CSMSRootCertificate |
| Test case Id | TC_M_12_CSMS |
| Use case Id(s) | M03 |
| Requirement(s) | M03.FR.01 |
| System under test | CSMS |
| Description | The CSMS is able to retrieve the certificates installed at the Charging Station using the GetInstalledCertificateIdsRequest message. It supports all available hash algorithms, including SHA256, SHA384, and SHA512. |
| Purpose | To verify if the CSMS is able to retrieve the hashData from all CSMSRootCertificates stored at the Charging Station, using all available hash algorithms, including SHA256, SHA384, and SHA512. |
| Prerequisite(s) | N/a |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>GetInstalledCertificates</i> for certificateType <i>CSMSRootCertificate</i> . The Test System responds with data hashed with SHA256. | |
| 2. Execute Reusable State <i>GetInstalledCertificates</i> for certificateType <i>CSMSRootCertificate</i> . The Test System responds with data hashed with SHA384. | |
| 3. Execute Reusable State <i>GetInstalledCertificates</i> for certificateType <i>CSMSRootCertificate</i> . The Test System responds with data hashed with SHA512. | |

| |
|--|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_M_13_CSMS: Retrieve certificates from Charging Station - ManufacturerRootCertificate

| | |
|-------------------|--|
| Test case name | Retrieve certificates from Charging Station - ManufacturerRootCertificate |
| Test case Id | TC_M_13_CSMS |
| Use case Id(s) | M03 |
| Requirement(s) | M03.FR.01 |
| System under test | CSMS |
| Description | The CSMS is able to retrieve the certificates installed at the Charging Station using the GetInstalledCertificateIdsRequest message. |
| Purpose | To verify if the CSMS is able to retrieve the hashData from all ManufacturerRootCertificate stored at the Charging Station. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>GetInstalledCertificates</i> for certificateType <i>ManufacturerRootCertificate</i> | |

| |
|-----------------------------------|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_M_14_CSMS: Retrieve certificates from Charging Station - V2GRootCertificate

| | |
|--------------------------|--|
| Test case name | Retrieve certificates from Charging Station - V2GRootCertificate |
| Test case Id | TC_M_14_CSMS |
| Use case Id(s) | M03 |
| Requirement(s) | M03.FR.01 |
| System under test | CSMS |
| Description | The CSMS is able to retrieve the certificates installed at the Charging Station using the GetInstalledCertificateIdsRequest message. |
| Purpose | To verify if the CSMS is able to retrieve the hashData from all V2GRootCertificate stored at the Charging Station. |
| Prerequisite(s) | N/a |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>GetInstalledCertificates</i> for certificateType V2GRootCertificate | |

| |
|--|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_M_15_CSMS: Retrieve certificates from Charging Station - V2GCertificateChain

| | |
|--------------------------|---|
| Test case name | Retrieve certificates from Charging Station - V2GCertificateChain |
| Test case Id | TC_M_15_CSMS |
| Use case Id(s) | M03 |
| Requirement(s) | M03.FR.01,M03.FR.05 |
| System under test | CSMS |
| Description | The CSMS is able to retrieve the certificates installed at the Charging Station using the GetInstalledCertificateIdsRequest message. |
| Purpose | To verify if the CSMS is able to retrieve the hashData from all certificates that are part of a V2GCertificateChain stored at the Charging Station. |
| Prerequisite(s) | N/a |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>GetInstalledCertificates</i> for certificateType V2GCertificateChain | |

| |
|--|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_M_16_CSMS: Retrieve certificates from Charging Station - MORootCertificate

| | |
|-------------------|--|
| Test case name | Retrieve certificates from Charging Station - MORootCertificate |
| Test case Id | TC_M_16_CSMS |
| Use case Id(s) | M03 |
| Requirement(s) | M03.FR.01 |
| System under test | CSMS |
| Description | The CSMS is able to retrieve the certificates installed at the Charging Station using the GetInstalledCertificateIdsRequest message. |
| Purpose | To verify if the CSMS is able to retrieve the hashData from all MORootCertificate stored at the Charging Station. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>GetInstalledCertificates</i> for certificateType <i>MORootCertificate</i> | |

| |
|-----------------------------------|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_M_17_CSMS: Retrieve certificates from Charging Station - CSMSRootCertificate & ManufacturerRootCertificate

| | |
|-------------------|--|
| Test case name | Retrieve certificates from Charging Station - CSMSRootCertificate & ManufacturerRootCertificate |
| Test case Id | TC_M_17_CSMS |
| Use case Id(s) | M03 |
| Requirement(s) | M03.FR.01 |
| System under test | CSMS |
| Description | The CSMS is able to retrieve the certificates installed at the Charging Station using the GetInstalledCertificateIdsRequest message. |
| Purpose | To verify if the CSMS is able to retrieve the hashData from all CSMSRootCertificates and ManufacturerRootCertificate stored at the Charging Station. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|------|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Execute Reusable State <i>GetInstalledCertificates</i> for certificateType <i>CSMSRootCertificate</i> AND <i>ManufacturerRootCertificate</i> | |

| |
|-----------------------------------|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

TC_M_18_CSMS: Retrieve certificates from Charging Station - All certificateTypes

| | |
|-------------------|--|
| Test case name | Retrieve certificates from Charging Station - All certificateTypes |
| Test case Id | TC_M_18_CSMS |
| Use case Id(s) | M03 |
| Requirement(s) | M03.FR.01 |
| System under test | CSMS |
| Description | The CSMS is able to retrieve the certificates installed at the Charging Station using the GetInstalledCertificateIdsRequest message. |
| Purpose | To verify if the CSMS is able to retrieve the hashData from all Root CA and V2GCertificateChain certificates stored at the Charging Station. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action:</u> Trigger the CSMS to send a <code>GetInstalledCertificateIdsRequest</code> without <code>certificateType</code> . | |
| 2. The Test System responds with a GetInstalledCertificateIdsResponse With status is <i>Accepted</i> certificateHashDataChain contains <i><The hashData of all certificates stored at the Test System truststore></i> | 1. The CSMS sends a GetInstalledCertificateIdsRequest |

| |
|---|
| Tool validations |
| * Step 1: Message: GetInstalledCertificateIdsRequest - certificateType is omitted |
| Post scenario validations: N/a |

TC_M_19_CSMS: Retrieve certificates from Charging Station - No matching certificate found

| | |
|-------------------|---|
| Test case name | Retrieve certificates from Charging Station - No matching certificate found |
| Test case Id | TC_M_19_CSMS |
| Use case Id(s) | M03 |
| Requirement(s) | M03.FR.01,M03.FR.02 |
| System under test | CSMS |
| Description | The CSMS is able to retrieve the certificates installed at the Charging Station using the <code>GetInstalledCertificateIdsRequest</code> message. |
| Purpose | To verify if the CSMS is able to handle a response from the Charging Station indicating it was not able to find a certificate for the requested criteria. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action:</u> <i>Trigger the CSMS to send a <code>GetInstalledCertificateIdsRequest</code> with <code>certificateType ManufacturerRootCertificate</code>.</i> | |
| 2. The Test System responds with a <code>GetInstalledCertificateIdsResponse</code> With status is <i>NotFound</i> certificateHashDataChain is omitted. | 1. The CSMS sends a <code>GetInstalledCertificateIdsRequest</code> |

| |
|---|
| Tool validations |
| * Step 1: Message: <code>GetInstalledCertificateIdsRequest</code> - certificateType is <i>ManufacturerRootCertificate</i> |
| Post scenario validations: N/a |

TC_M_20_CSMS: Delete a certificate from a Charging Station - Success

| | |
|-------------------|--|
| Test case name | Delete a certificate from a Charging Station - Success |
| Test case Id | TC_M_20_CSMS |
| Use case Id(s) | M04 |
| Requirement(s) | M04.FR.01,M04.FR.07 |
| System under test | CSMS |
| Description | The CSMS is able to request the Charging Station to delete an installed certificate using the DeleteCertificateRequest message, using all available hash algorithms, including SHA256, SHA384, and SHA512. |
| Purpose | To verify if CSMS is able to request a Charging Station to delete an installed certificate, using all available hash algorithms, including SHA256, SHA384, and SHA512. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|--|--|
| Charging Station | CSMS |
| 1. <i>CertificateInstalled</i> with certificateType <i>CSMSRootCertificate</i> . | |
| <u>Manual Action</u> : Request the CSMS to send a <i>DeleteCertificateRequest</i> . | |
| 3. The Test System responds with a GetInstalledCertificateIdsResponse With status is <i>Accepted</i> certificateHashDataChain contains an entry with following values: certificateHashDataChain[0].certificateType is <i>CSMSRootCertificate</i> certificateHashDataChain[0].certificateHashData.hashAlgorithm is <i>SHA256</i> | 2. The CSMS sends a GetInstalledCertificateIdsRequest |
| 5. The Test System responds with a DeleteCertificateResponse With status is <i>Accepted</i> | 4. The CSMS sends a DeleteCertificateRequest |
| <u>Note(s)</u> : - Steps 1 - 5 will be repeated for each hash algorithm (<i>SHA256</i> , <i>SHA384</i> , <i>SHA512</i>). | |

| |
|---|
| Tool validations |
| * Step 2: Message: GetInstalledCertificateIdsRequest - certificateType contains <i>CSMSRootCertificate</i> OR is <i>omitted</i> . |
| * Step 4: Message: DeleteCertificateRequest - certificateHashData is <i><Returned certificateHashData at Step 3></i> . |
| Post scenario validations: N/a |

TC_M_21_CSMS: Delete a certificate from a Charging Station - Failed

| | |
|-------------------|---|
| Test case name | Delete a certificate from a Charging Station - Failed |
| Test case Id | TC_M_21_CSMS |
| Use case Id(s) | M04 |
| Requirement(s) | M04.FR.01,M04.FR.07 |
| System under test | CSMS |
| Description | The CSMS is able to request the Charging Station to delete an installed certificate using the DeleteCertificateRequest message. |
| Purpose | To verify if CSMS is able to handle a Charging Station that fails to delete an installed certificate. |
| Prerequisite(s) | N/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): <i>CertificateInstalled</i> with certificateType <i>CSMSRootCertificate</i> . |

| Main (Test scenario) | |
|--|--|
| Charging Station | CSMS |
| <u>Manual Action</u> : Request the CSMS to send a DeleteCertificateRequest. | |
| 2. The Test System responds with a GetInstalledCertificateIdsResponse With status is <i>Accepted</i> certificateHashDataChain contains an entry with following values: certificateHashDataChain[0].certificateType is <i>CSMSRootCertificate</i> certificateHashDataChain[0].certificateHashData.hashAlgorithm is <i>SHA256</i> | 1. The CSMS sends a GetInstalledCertificateIdsRequest |
| 4. The Test System responds with a DeleteCertificateResponse With status is <i>Failed</i> | 3. The CSMS sends a DeleteCertificateRequest |

| |
|--|
| Tool validations |
| * Step 1: Message: GetInstalledCertificateIdsRequest - certificateType contains <i>CSMSRootCertificate</i> OR is <i>omitted</i> . |
| * Step 3: Message: DeleteCertificateRequest - certificateHashData contains <i><Returned certificateHashData at Step 2></i> . |
| Post scenario validations: N/a |

TC_M_24_CSMS: Get Charging Station Certificate status - Success

| | |
|-------------------|---|
| Test case name | Get Charging Station Certificate status - Success |
| Test case Id | TC_M_24_CSMS |
| Use case Id(s) | M06 |
| Requirement(s) | M06.FR.01,M06.FR.02,M06.FR.03,M06.FR.08,M06.FR.09 |
| System under test | CSMS |
| Description | The Charging Station is able to request the CSMS to get the status of a (V2G) Charging Station certificate. |
| Purpose | To verify if the CSMS is able to provide the status of a requested (V2G) Charging Station certificate. |
| Prerequisite(s) | N/a |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State(s):

N/a

Main (Test scenario)

| Charging Station | CSMS |
|--|--|
| 1. The Test System sends one or more subsequent GetCertificateStatusRequests With ocspRequestData contains <i><hashes from configured (V2G) certificate chain SubCA's></i> | 2. The CSMS responds with a GetCertificateStatusResponse |

Tool validations

Step 2:

Message: **GetCertificateStatusResponse****status** *Accepted***ocspResult** *<OCSPResponse class as defined in IETF RFC 6960. DER encoded (as defined in IETF RFC 6960), and then base64 encoded.>*

Post scenario validations:

N/a

TC_M_26_CSMS: Certificate Installation EV - Success

| | |
|--------------------------|---|
| Test case name | Certificate Installation EV - Success |
| Test case Id | TC_M_26_CSMS |
| Use case Id(s) | M01 |
| Requirement(s) | M01.FR.01 |
| System under test | CSMS |
| Description | The EV initiates installing a new certificate. The Charging Station forwards the request for a new certificate to the CSMS. |
| Purpose | To verify if the CSMS is able to return the Raw CertificateInstallationRes response for the EV to the Charging Station. |
| Prerequisite(s) | N/a |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Test System sends a Get15118EVCertificateRequest With action Install | 2. The CSMS responds with a Get15118EVCertificateResponse |

| |
|--|
| Tool validations |
| * Step 2: Message: Get15118EVCertificateResponse - status <i>Accepted</i> - exiResponse <i><Raw CertificateInstallationRes response for the EV, Base64 encoded.></i> |
| Post scenario validations: N/a |

TC_M_28_CSMS: Certificate Update EV - Success

| | |
|--------------------------|---|
| Test case name | Certificate Update EV - Success |
| Test case Id | TC_M_28_CSMS |
| Use case Id(s) | M02 |
| Requirement(s) | M02.FR.01 |
| System under test | CSMS |
| Description | The EV initiates updating the existing certificate. The Charging Station forwards the update request to the CSMS. |
| Purpose | To verify if the CSMS is able to return the Raw CertificateInstallationRes response for the EV to the Charging Station. |
| Prerequisite(s) | N/a |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Test System sends a Get15118EVCertificateRequest With action Update | 2. The CSMS responds with a Get15118EVCertificateResponse |

| |
|--|
| Tool validations |
| <p>* Step 2:</p> <p>Message: Get15118EVCertificateResponse</p> <ul style="list-style-type: none"> - status <i>Accepted</i> - exiResponse <i><Raw CertificateInstallationRes response for the EV, Base64 encoded.></i> |
| Post scenario validations: N/a |

N Diagnostics

TC_N_01_CSMS: Get Monitoring Report - with monitoringCriteria

| | |
|-------------------|---|
| Test case name | Get Monitoring Report - with monitoringCriteria |
| Test case Id | TC_N_01_CSMS |
| Use case Id(s) | N02 |
| Requirement(s) | N02.FR.05, N02.FR.10 |
| System under test | CSMS |
| Description | CSMS requests a report of monitors that match the component criteria. |
| Purpose | To test that CSMS supports requesting a monitoring report for the component criteria and that it handles an empty result set. |
| Prerequisite(s) | CS has implemented device model monitoring and MonitoringCtrlr.Enabled = true. |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: CSMS requests ClearVariableMonitoring ItemsPerMessage from CS. |
| Reusable State(s): N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| Manually instruct CSMS to get a report of monitors for: - all <i>DeltaMonitoring</i> | |
| 2. Test System responds with: GetMonitoringReportResponse with: Status <i>EmptyResultSet</i> | 1. CSMS sends GetMonitoringReportRequest |
| Manually instruct CSMS to get a report of monitors for: - all <i>ThresholdMonitoring</i> | |
| 4. Test System responds with: GetMonitoringReportResponse with: Status <i>Accepted</i> | 3. CSMS sends GetMonitoringReportRequest |
| 5. Test System responds with: NotifyMonitoringReportRequest | 6. CSMS sends NotifyMonitoringReportResponse |
| Step 5 and 6 are repeated as often as needed to report all configuration variables. | |

| |
|---|
| Tool validations |
| * Step 1: Message: GetMonitoringReportRequest - monitoringCriteria = <i>DeltaMonitoring</i> - componentVariable is omitted. |
| * Step 3: Message: GetMonitoringReportRequest - monitoringCriteria = <i>ThresholdMonitoring</i> - componentVariable is omitted. |
| Post scenario validations: Check that CSMS shows the <i>Threshold</i> monitors. |

TC_N_02_CSMS: Get Monitoring Report - with component/variable

| | |
|--------------------------|---|
| Test case name | Get Monitoring Report - with component/variable |
| Test case Id | TC_N_02_CSMS |
| Use case Id(s) | N02 |
| Requirement(s) | N02.FR.05, N02.FR.10 |
| System under test | CSMS |
| Description | CSMS requests a report of monitors that match the the given list of components and variables. |
| Purpose | To test that CSMS supports requesting a monitoring report for a given component and variable and that it handles an empty result set. |
| Prerequisite(s) | CS has implemented device model monitoring and MonitoringCtrlr.Enabled = true. |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: CSMS requests ClearVariableMonitoring ItemsPerMessage from CS. |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| Manually instruct CSMS to get a report of monitors for: - the variable Power of ChargingStation | |
| 2. Test System responds with: GetMonitoringReportResponse with: Status EmptyResultSet | 1. CSMS sends GetMonitoringReportRequest |
| Manually instruct CSMS to get a report of monitors for: - the variable AvailabilityState of EVSE #1. | |
| 4. Test System responds with: GetMonitoringReportResponse with: Status Accepted | 3. CSMS sends GetMonitoringReportRequest |
| 5. Test System responds with: NotifyMonitoringReportRequest | 6. CSMS sends NotifyMonitoringReportResponse |
| Step 5 and 6 are repeated as often as needed to report all configuration variables. | |

| |
|--|
| Tool validations |
| <p>* Step 1:</p> <p>Message: GetMonitoringReportRequest</p> <ul style="list-style-type: none"> - componentVariable[0].component.name = "ChargingStation" - componentVariable[0].component.instance is omitted. - componentVariable[0].variable.name = "Power" - componentVariable[0].variable.instance is omitted. - monitoringCriteria is omitted. |

| Tool validations |
|---|
| <p>* Step 3:</p> <p>Message: GetMonitoringReportRequest</p> <ul style="list-style-type: none">- componentVariable[1].component.name = "EVSE"- componentVariable[1].component.instance is omitted.- componentVariable[1].component.evse.id = 1- componentVariable[1].variable.name = "AvailabilityState"- componentVariable[1].variable.instance is omitted.- monitoringCriteria is omitted. |
| <p>Post scenario validations:</p> <p>Check that CSMS shows the monitor for AvailabilityState for EVSE #1.</p> |

TC_N_03_CSMS: Get Monitoring Report - with component criteria and component/variable

| | |
|-------------------|---|
| Test case name | Get Monitoring Report - with component criteria and component/variable |
| Test case Id | TC_N_03_CSMS |
| Use case Id(s) | N02 |
| Requirement(s) | N02.FR.05, N02.FR.10 |
| System under test | CSMS |
| Description | CSMS requests a report of monitors that match both the component criteria and the given list of components and variables. |
| Purpose | To test that CSMS supports requesting a monitoring report for both the component criteria and a given component and variable and that it handles an empty result set. |
| Prerequisite(s) | CS has implemented device model monitoring and MonitoringCtrlr.Enabled = true. |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: CSMS requests ClearVariableMonitoring ItemsPerMessage from CS. |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| Manually instruct CSMS to get a report of monitors for: - all <i>DeltaMonitoring</i> - and the variable <i>AvailabilityState</i> for EVSE #1. | |
| 2. Test System responds with: GetMonitoringReportResponse with: Status <i>EmptyResultSet</i> | 1. CSMS sends GetMonitoringReportRequest |
| Manually instruct CSMS to get a report of monitors for: - all <i>ThresholdMonitoring</i> - and the variable <i>Power</i> of ChargingStation. | |
| 4. Test System responds with: GetMonitoringReportResponse with: Status <i>Accepted</i> | 3. CSMS sends GetMonitoringReportRequest |
| 5. Test System responds with: NotifyMonitoringReportRequest | 6. CSMS sends NotifyMonitoringReportResponse |
| Step 5 and 6 are repeated as often as needed to report all configuration variables. | |

| |
|------------------|
| Tool validations |
| |

Tool validations

* Step 1:

Message: **GetMonitoringReportRequest**

- **monitoringCriteria** = *DeltaMonitoring*
- **componentVariable[0].component.name** = "EVSE"
- **componentVariable[0].component.evse.id** = <configured evseld>
- **componentVariable[0].variable.name** = "AvailabilityState"

* Step 3:

Message: **GetMonitoringReportRequest**

- **monitoringCriteria** = *ThresholdMonitoring*
- **componentVariable[0].component.name** = "ChargingStation"
- **componentVariable[0].variable.name** = "Power"

Post scenario validations:

Check that CSMS shows the *Threshold* monitors for Power for ChargingStation.

TC_N_60_CSMS: Get Monitoring Report - with component criteria and list of components/variables

| | |
|-------------------|---|
| Test case name | Get Monitoring Report - with component criteria and list of components/variables |
| Test case Id | TC_N_60_CSMS |
| Use case Id(s) | N02 |
| Requirement(s) | N02.FR.05, N02.FR.10 |
| System under test | CSMS |
| Description | CSMS requests a report of monitors that match both the component criteria and the given list of components and variables. |
| Purpose | To test that CSMS supports requesting a monitoring report for both the component criteria and a given list of components and optionally with variables and that it handles an empty result set. |
| Prerequisite(s) | CS has implemented device model monitoring and MonitoringCtrlr.Enabled = true. |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: CSMS requests ClearVariableMonitoring ItemsPerMessage from CS. |
| Reusable State(s): N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| Manually instruct CSMS to get a report of monitors for: - all ThresholdMonitoring - and the variable Power of both ChargingStation and EVSE #1. | |
| 2. Test System responds with: GetMonitoringReportResponse with: Status EmptyResultSet | 1. CSMS sends GetMonitoringReportRequest |
| Manually instruct CSMS to get a report of monitors for: - all DeltaMonitoring - and the variable AvailabilityState of both ChargingStation and EVSE #1. | |
| 4. Test System responds with: GetMonitoringReportResponse with: Status Accepted | 3. CSMS sends GetMonitoringReportRequest |
| 5. Test System responds with: NotifyMonitoringReportRequest | 6. CSMS sends NotifyMonitoringReportResponse |
| Step 5 and 6 are repeated as often as needed to report all configuration variables. | |

| Tool validations |
|--|
| <p>* Step 1:</p> <p>Message: GetMonitoringReportRequest</p> <ul style="list-style-type: none">- monitoringCriteria is <i>DeltaMonitoring</i>- componentVariable[0].component.name = "<i>ChargingStation</i>"- componentVariable[0].variable.name = "<i>AvailabilityState</i>"- componentVariable[1].component.name = "<i>EVSE</i>"- componentVariable[1].component.evse.id = <configured evseId>- componentVariable[1].variable.name = "<i>AvailabilityState</i>" <p>* Step 3:</p> <p>Message: GetMonitoringReportRequest</p> <ul style="list-style-type: none">- monitoringCriteria = <i>ThresholdMonitoring</i>- componentVariable[0].component.name = "<i>ChargingStation</i>"- componentVariable[0].variable.name = "<i>AvailabilityState</i>"- componentVariable[1].component.name = "<i>EVSE</i>"- componentVariable[1].component.evse.id = <configured evseId>- componentVariable[1].variable.name = "<i>AvailabilityState</i>" |
| <p>Post scenario validations:</p> <p>Check that CSMS shows the <i>Delta</i> monitors for <i>AvailabilityState</i> for both <i>ChargingStation</i> and <i>EVSE</i> #1.</p> |

TC_N_05_CSMS: Set Monitoring Base - success

| | |
|-------------------|---|
| Test case name | Set Monitoring Base - success |
| Test case Id | TC_N_05_CSMS |
| Use case Id(s) | N03 |
| Requirement(s) | N03.FR.03, N03.FR.04, N03.FR.05 |
| System under test | CSMS |
| Description | CSMS sends a SetMonitoringBaseRequest for <i>All</i> , <i>FactoryDefault</i> and <i>HardWiredOnly</i> . |
| Purpose | To test that CSMS supports all three monitoring base types. |
| Prerequisite(s) | CS has implemented device model monitoring and MonitoringCtrlr.Enabled = true. |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. Test System responds with: SetMonitoringBaseResponse | <i>Instruct CSMS to set a monitoring base of _All._</i> 1. CSMS sends SetMonitoringBaseRequest |
| 4. Test System responds with: SetMonitoringBaseResponse | <i>Instruct CSMS to set a monitoring base of _FactoryDefault._</i> 3. Test System sends SetMonitoringBaseRequest |
| 6. The Test System responds with: SetMonitoringBaseResponse | <i>Instruct CSMS to set a monitoring base of _HardWiredOnly._</i> 5. Test System sends SetMonitoringBaseRequest |

| |
|---|
| Tool validations |
| * Step 1 Message: SetMonitoringBaseRequest - monitoringBase = <i>All</i> |
| * Step 3 Message: SetMonitoringBaseRequest - monitoringBase = <i>FactoryDefault</i> |
| * Step 6 Message: SetMonitoringBaseRequest - monitoringBase = <i>HardWiredOnly</i> |
| Post scenario validations: N/A |

TC_N_08_CSMS: Set Variable Monitoring - One SetMonitoringData element

| | |
|-------------------|--|
| Test case name | Set Variable Monitoring - One SetMonitoringData element |
| Test case Id | TC_N_08_CSMS |
| Use case Id(s) | N04 |
| Requirement(s) | N04.FR.01, N04.FR.02, N04.FR.17 |
| System under test | CSMS |
| Description | CSMS sends a request to activate monitoring on one variable. |
| Purpose | To test that CSMS supports setting monitoring on one variable. |
| Prerequisite(s) | CS has implemented device model monitoring and MonitoringCtrlr.Enabled = true. |

Before (Preparations)

Configuration State:

This test case activates monitoring on the following variable:

- Component "EVSE", evse <Configured evseld>, variable "AvailabilityState", monitor type *Delta*

It assumes, that no monitor is active on this variable prior to the test.

Note 1: this is a required variable for which a monitor can be expected to exist or it can be configured.

Note 2: Any other component/variable combination that supports monitoring could also be used for this test case.

Memory State:

N/a

Reusable State(s):

N/a

Main (Test scenario)

| Charging Station | CSMS |
|---|---|
| <p>2. Test System responds with: Message: SetVariableMonitoringResponse with setMonitoringResult[0].status = Accepted</p> | <p>1. Request CSMS to install monitors on: - EVSE #<Configured evseld>, AvailabilityState, <i>Delta</i>, severity 8</p> |

Tool validations

* Step 1:

1. CSMS sends **SetVariableMonitoringRequest** with:

- **setMonitoringData[0].value** = 1, ← recommended value for *Delta* monitor
- **setMonitoringData[0].type** = *Delta*,
- **setMonitoringData[0].severity** = 8,
- **setMonitoringData[0].component.name** = "EVSE"
- **setMonitoringData[0].component.evse.id** = <Configured evseld>
- **setMonitoringData[0].variable.name** = "AvailabilityState"

Post scenario validations:

N/A

TC_N_09_CSMS: Set Variable Monitoring - Multiple elements on different component and variable

| | |
|-------------------|---|
| Test case name | Set Variable Monitoring - Multiple elements on different component and variable |
| Test case Id | TC_N_09_CSMS |
| Use case Id(s) | N04 |
| Requirement(s) | N04.FR.01, N04.FR.02, N04.FR.17 |
| System under test | CSMS |
| Description | CSMS sends a request to activate monitors on different variables. |
| Purpose | To test that CSMS supports setting of multiple monitors on different variables. |
| Prerequisite(s) | CS has implemented device model monitoring and MonitoringCtrlr.Enabled = true. |

| |
|---|
| Before (Preparations) |
| <p>Configuration State:</p> <p>This test case activates monitors on the following variables:</p> <ul style="list-style-type: none"> - Component "EVSE", evse <Configured evseld>, variable "AvailabilityState", monitor type <i>Delta</i> - Component "ChargingStation", variable "AvailabilityState", monitor type <i>Delta</i> <p>It assumes, that no monitor is active on these variables prior to the test.</p> <p><i>Note 1: these are required variables for which a monitor can be expected to exist or it can be configured.</i></p> <p><i>Note 2: Any other component/variable combination that supports monitoring could also be used for this test case.</i></p> <p>Memory State: N/a</p> <p>Reusable State(s): N/a</p> |

| Main (Test scenario) | |
|---|---|
| Charging Station | CSMS |
| 2. Test System responds with: Message: SetVariableMonitoringResponse with setMonitoringResult[0].status = <i>Accepted</i> | 1. Request CSMS to install monitors on: - EVSE #<Configured evseld>, AvailabilityState, Delta, severity 8 - ChargingStation, AvailabilityState, Delta, severity 8 |

| |
|--|
| Tool validations |
| <p>* Step 1:</p> <p>1. CSMS sends SetVariableMonitoringRequest with:</p> <ul style="list-style-type: none"> - setMonitoringData[0].value = 1, ← recommended value for <i>Delta</i> monitor - setMonitoringData[0].type = Delta, - setMonitoringData[0].severity = 8, - setMonitoringData[0].component.name = "EVSE" - setMonitoringData[0].component.evse.id = <Configured evseld> - setMonitoringData[0].variable.name = "AvailabilityState" <ul style="list-style-type: none"> - setMonitoringDate[1].value = 1, - setMonitoringDate[1].type = Delta, - setMonitoringDate[1].severity = 8, - setMonitoringDate[1].component.name = "ChargingStation" - setMonitoringDate[1].variable.name = "AvailabilityState" <p>Post scenario validations: N/A</p> |

TC_N_16_CSMS: Set Monitoring Level - Success

| | |
|-------------------|---|
| Test case name | Set Monitoring Level - Success |
| Test case Id | TC_N_16_CSMS |
| Use case Id(s) | N05 |
| Requirement(s) | N05.FR.01 |
| System under test | CSMS |
| Description | CSMS sets a monitoring level. |
| Purpose | To test that CSMS supports setting of a monitoring level. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|---|---|
| 2. Test System responds with: SetMonitoringLevelResponse with Status is <i>Accepted</i> | 1. <i>Instruct CSMS to set a monitoring level with severity = _4</i> |

| |
|---|
| Tool validations |
| * Step 1: Message: SetMonitoringLevelRequest with: severity = 4 |
| Post scenario validations: N/A |

TC_N_17_CSMS: Set Monitoring Level - Out of range

| | |
|-------------------|---|
| Test case name | Set Monitoring Level - Out of range |
| Test case Id | TC_N_17_CSMS |
| Use case Id(s) | N05 |
| Requirement(s) | N05.FR.02 |
| System under test | CSMS |
| Description | CSMS sets a monitoring level. |
| Purpose | To test that CSMS supports the rejection of setting of a monitoring level. |
| Prerequisite(s) | The Test System will always reject the message, but normally this would only occur if the set severity level is out of range. |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. Test System responds with: SetMonitoringLevelResponse with Status is <i>Rejected</i> | 1. <i>Instruct CSMS to set a monitoring level with severity = 4</i> |

| |
|---|
| Tool validations |
| * Step 1: Message: SetMonitoringLevelRequest with: severity = 4 |
| Post scenario validations: N/A |

TC_N_18_CSMS: Clear Monitoring - Too many elements

| | |
|-------------------|--|
| Test case name | Clear Monitoring - Too many elements |
| Test case Id | TC_N_18_CSMS |
| Use case Id(s) | N06 |
| Requirement(s) | N06.FR.04 |
| System under test | CSMS |
| Description | CSMS is requested to clear more monitors than allowed in one request. |
| Purpose | To test that CSMS does not exceed the <code>ItemsPerMessageClearVariableMonitoring</code> amount of monitors in one request. |
| Prerequisite(s) | CS has implemented device model monitoring and <code>MonitoringCtrlr.Enabled = true</code> . |

Before (Preparations)

Configuration State:

This test requests the value of `ItemsPerMessageClearVariableMonitoring` and then instructs the CSMS to clear (at least) one more monitor than allowed by this value. This value is 'read-only', so it cannot be manipulated in the test. As a consequence, if the Charging Station supports more monitor ids in the list, than can be set by the CSMS, then this cannot be tested.

Memory State:

N/a

Reusable State(s):

N/a

Main (Test scenario)

| | |
|---|---|
| 2. The Test System responds with: GetVariablesResponse | 1. Instruct CSMS to send <i>GetVariablesRequest</i> with: Component.name <i>MonitoringCtrlr</i> Variable.name <i>ItemsPerMessage</i> Variable.instance <i>ClearVariableMonitoring</i> . |
| 4. The Test System responds with: ClearVariableMonitoringResponse | 3. Instruct CSMS to clear more monitors than allowed in <i>ItemsPerMessage</i> . ClearVariableMonitoringRequest with a list of ids <i>Note: these monitor ids do not have to exist.</i> |

Tool validations

* Step 1:

Message: Two or more **ClearVariableMonitoringRequest**, so that the maximum number of `ItemsPerMessageClearVariableMonitoring` **ids** is never exceeded.

Test System will reply with a **ClearVariableMonitoringResponse** for each **ClearVariableMonitoringRequest**, but the content of the responses is irrelevant for the test.

Post scenario validations:

N/A

TC_N_21_CSMS: Alert Event - HardWiredMonitor

| | |
|-------------------|--|
| Test case name | Alert Event - HardWiredMonitor |
| Test case Id | TC_N_21_CSMS |
| Use case Id(s) | N07 |
| Requirement(s) | N07.FR.03 |
| System under test | CSMS |
| Description | Charging Station sends an NotifyEventRequest for a HardWiredMonitor. |
| Purpose | To test that the CSMS is able to handle a HardWiredMonitor. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. Test System sends NotifyEventRequest message with eventNotificationType = <i>HardWiredMonitor</i> | 2. CSMS returns NotifyEventResponse message. |

| |
|---|
| Tool validations |
| * Step 2: Message: NotifyEventResponse with empty body. |
| Post scenario validations: N/A |

TC_N_24_CSMS: Set Variable Monitoring - Periodic event

| | |
|-------------------|---|
| Test case name | Set Variable Monitoring - Periodic event |
| Test case Id | TC_N_24_CSMS |
| Use case Id(s) | N08 |
| Requirement(s) | N08.FR.02 |
| System under test | CSMS |
| Description | Charging Station sends a periodic NotifyEventRequest. |
| Purpose | To test that CSMS returns a NotifyEventResponse. <i>Note: this is identical to TC_N_21_CSMS, only with a periodic event.</i> |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| Tester makes Test System send a NotifyEventRequest message. | |
| 1. Test System sends NotifyEventRequest message. | 2. CSMS returns NotifyEventResponse message. |
| <u>Note(s)</u> : - Step 1 and 2 will be repeated n times | |

| |
|---|
| Tool validations |
| * Step 2: Message: NotifyEventResponse with empty body. |
| Post scenario validations: N/A |

TC_N_25_CSMS: Retrieve Log Information - Diagnostics Log - Success

| | |
|-------------------|--|
| Test case name | Retrieve Log Information - Diagnostics Log - Success |
| Test case Id | TC_N_25_CSMS |
| Use case Id(s) | N01 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | This test case covers the functionality of getting log information from a Charging Station. The CSMS can request a Charging Station to upload a file with log information to a given location (URL). The format of this log file is not prescribed. The Charging Station successfully uploads a log file and gives information about the status of the upload by sending status notifications to the CSMS. |
| Purpose | To verify if the CSMS is able to request a charging station to successfully upload a log as described at the OCPP specification. |
| Prerequisite(s) | Charging Station has log information available. |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State:

N/a

Main (Test scenario)

| Charging Station | CSMS |
|---|--|
| 2. The Test System responds with a GetLogResponse with status Accepted | 1. The CSMS sends a GetLogRequest |
| 3. The Test System sends a LogStatusNotificationRequest with - status Uploading - requestId Same Id as the GetLogRequest | 4. The CSMS responds with a LogStatusNotificationResponse . |
| 5. The Test System sends a LogStatusNotificationRequest with - status Uploaded - requestId Same Id as the GetLogRequest | 6. The CSMS responds with a LogStatusNotificationResponse . |

Tool validations

* Step 1:

Message **GetLogRequest**- **logType DiagnosticsLog**

Post scenario validations:

- N/a

TC_N_27_CSMS: Get Customer Information - Accepted + data

| | |
|-------------------|--|
| Test case name | Get Customer Information - Accepted + data |
| Test case Id | TC_N_27_CSMS |
| Use case Id(s) | N09 |
| Requirement(s) | N09.FR.01, N09.FR.04 |
| System under test | CSMS |
| Description | The CSMS sends a message to the Charging Station to retrieve IdToken customer information, for example to be compliant with local privacy laws. The Charging Station notifies the CSMS by sending one or more reports. |
| Purpose | To verify if the CSMS sends the request correctly and responds on the notifies as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a CustomerInformationResponse with status Accepted | 1. The CSMS sends a CustomerInformationRequest |
| 3. The Test System sends a NotifyCustomerInformationRequest | 4. The CSMS responds with a NotifyCustomerInformationResponse . |

| |
|--|
| Tool validations |
| <p>* Step 1:</p> <p>Message CustomerInformationRequest</p> <ul style="list-style-type: none"> - report true - idToken.idToken <Configured valid_idtoken_idtoken> - idToken.type <Configured valid_idtoken_type> |
| <p>Post scenario validations:</p> <ul style="list-style-type: none"> - N/a |

TC_N_28_CSMS: Get Customer Information - Accepted + no data

| | |
|-------------------|--|
| Test case name | Get Customer Information - Accepted + no data |
| Test case Id | TC_N_28_CSMS |
| Use case Id(s) | N09 |
| Requirement(s) | N09.FR.01, N09.FR.04 |
| System under test | CSMS |
| Description | The CSMS sends a message to the Charging Station to retrieve IdToken customer information, for example to be compliant with local privacy laws. The Charging Station notifies the CSMS by sending one or more reports. |
| Purpose | To verify if the CSMS sends the request correctly and responds on the notifies as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a CustomerInformationResponse with status Accepted | 1. The CSMS sends a CustomerInformationRequest |
| 3. The Test System sends a NotifyCustomerInformationRequest | 4. The CSMS responds with a NotifyCustomerInformationResponse . |

| |
|--|
| Tool validations |
| <p>* Step 1:</p> <p>Message CustomerInformationRequest</p> <ul style="list-style-type: none"> - report true - idToken.idToken <Configured valid_idtoken_idtoken> - idToken.type <Configured valid_idtoken_type> |
| <p>Post scenario validations:</p> <ul style="list-style-type: none"> - N/a |

TC_N_29_CSMS: Get Customer Information - Not Accepted

| | |
|-------------------|--|
| Test case name | Get Customer Information - Not Accepted |
| Test case Id | TC_N_29_CSMS |
| Use case Id(s) | N09 |
| Requirement(s) | N09.FR.01, N09.FR.04 |
| System under test | CSMS |
| Description | The CSMS sends a message to the Charging Station to retrieve IdToken customer information, but the Charging Station rejects the request. |
| Purpose | To verify if the CSMS sends the request correctly as described at the OCPP specification, and can handle the Charging Station rejecting the request. |
| Prerequisite(s) | n/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a CustomerInformationResponse with status <i>Rejected</i> | 1. The CSMS sends a CustomerInformationRequest |

| |
|--|
| Tool validations |
| <p>* Step 1:</p> <p>Message CustomerInformationRequest</p> <ul style="list-style-type: none"> - report true - idToken.idToken <Configured valid_idtoken_idtoken> - idToken.type <Configured valid_idtoken_type> |
| <p>Post scenario validations:</p> <ul style="list-style-type: none"> - N/a |

TC_N_30_CSMS: Clear Customer Information - Clear and report + data

| | |
|-------------------|--|
| Test case name | Clear Customer Information - Clear and report + data |
| Test case Id | TC_N_30_CSMS |
| Use case Id(s) | N10 |
| Requirement(s) | N10.FR.08 |
| System under test | CSMS |
| Description | The CSMS sends a message to the Charging Station to clear (and retrieve) IdToken customer information, for example to be compliant with local privacy laws. The Charging Station notifies the CSMS by sending one or more reports. |
| Purpose | To verify if the CSMS sends the request correctly and responds on the notifies as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a CustomerInformationResponse with status Accepted | 1. The CSMS sends a CustomerInformationRequest |
| 3. The Test System sends a NotifyCustomerInformationRequest | 4. The CSMS responds with a NotifyCustomerInformationResponse . |

| |
|---|
| Tool validations |
| <p>* Step 1:</p> <p>Message CustomerInformationRequest</p> <ul style="list-style-type: none"> - report <i>true</i> - clear <i>true</i> - idToken.idToken <Configured valid_idtoken_idtoken> - idToken.type <Configured valid_idtoken_type> |
| <p>Post scenario validations:</p> <ul style="list-style-type: none"> - N/a |

TC_N_31_CSMS: Clear Customer Information - Clear and report + no data

| | |
|-------------------|--|
| Test case name | Clear Customer Information - Clear and report + no data |
| Test case Id | TC_N_31_CSMS |
| Use case Id(s) | N10 |
| Requirement(s) | N10.FR.08 |
| System under test | CSMS |
| Description | The CSMS sends a message to the Charging Station to clear (and retrieve) IdToken customer information, for example to be compliant with local privacy laws. The Charging Station notifies the CSMS by sending one or more reports. |
| Purpose | To verify if the CSMS sends the request correctly and responds on the notifies as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a CustomerInformationResponse with status Accepted | 1. The CSMS sends a CustomerInformationRequest |
| 3. The Test System sends a NotifyCustomerInformationRequest | 4. The CSMS responds with a NotifyCustomerInformationResponse . |

| |
|---|
| Tool validations |
| <p>* Step 1:</p> <p>Message CustomerInformationRequest</p> <ul style="list-style-type: none"> - report <i>true</i> - clear <i>true</i> - idToken.idToken <Configured valid_idtoken_idtoken> - idToken.type <Configured valid_idtoken_type> |
| <p>Post scenario validations:</p> <ul style="list-style-type: none"> - N/a |

TC_N_32_CSMS: Clear Customer Information - Clear and no report

| | |
|-------------------|--|
| Test case name | Clear Customer Information - Clear and no report |
| Test case Id | TC_N_32_CSMS |
| Use case Id(s) | N10 |
| Requirement(s) | N10.FR.08 |
| System under test | CSMS |
| Description | The CSMS sends a message to the Charging Station to clear IdToken customer information, for example to be compliant with local privacy laws. |
| Purpose | To verify if the CSMS sends the request correctly. |
| Prerequisite(s) | n/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| Main (Test scenario) | |
|--|--|
| Charging Station | CSMS |
| 2. The Test System responds with a CustomerInformationResponse with status <i>Accepted</i> | 1. The CSMS sends a CustomerInformationRequest |
| 3. The Test System sends a NotifyCustomerInformationRequest | 4. The CSMS responds with a NotifyCustomerInformationResponse |

| |
|--|
| Tool validations |
| <p>* Step 1:</p> <p>Message CustomerInformationRequest</p> <ul style="list-style-type: none"> - report <i>false</i> - clear <i>true</i> - idToken.idToken <Configured valid_idtoken_idtoken> - idToken.type <Configured valid_idtoken_type> |
| <p>Post scenario validations:</p> <ul style="list-style-type: none"> - N/a |

TC_N_62_CSMS: Clear Customer Information - Clear and report - customerIdentifier

| | |
|--------------------------|--|
| Test case name | Clear Customer Information - Clear and report - customerIdentifier |
| Test case Id | TC_N_62_CSMS |
| Use case Id(s) | N10 |
| Requirement(s) | N10.FR.08 |
| System under test | CSMS |
| Description | The CSMS sends a message to the Charging Station to clear (and retrieve) raw customer information, for example to be compliant with local privacy laws. The Charging Station notifies the CSMS by sending one or more reports. |
| Purpose | To verify if the CSMS sends the request correctly and responds to the notifications as described in the OCPP specification. |
| Prerequisite(s) | The CSMS supports retrieving / deleting CustomerInformation - CustomerIdentifier |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a CustomerInformationResponse with status Accepted | 1. The CSMS sends a CustomerInformationRequest |
| 3. The Test System sends a NotifyCustomerInformationRequest | 4. The CSMS responds with a NotifyCustomerInformationResponse |

| |
|--|
| Tool validations |
| <p>* Step 1:</p> <p>Message CustomerInformationRequest</p> <ul style="list-style-type: none"> - report <i>true</i> - clear <i>true</i> - customerIdentifier <i>"OpenChargeAlliance"</i> - idToken is omitted - customerCertificate is omitted |
| Post scenario validations: |
| - N/a |

TC_N_63_CSMS: Clear Customer Information - Clear and report - customerCertificate

| | |
|-------------------|--|
| Test case name | Clear Customer Information - Clear and report - customerCertificate |
| Test case Id | TC_N_63_CSMS |
| Use case Id(s) | N10 |
| Requirement(s) | N10.FR.08 |
| System under test | CSMS |
| Description | The CSMS sends a message to the Charging Station to clear (and retrieve) a customer certificate, for example to be compliant with local privacy laws. The Charging Station notifies the CSMS by sending one or more reports. |
| Purpose | To verify if the CSMS sends the request correctly and responds to the notifications as described in the OCPP specification. |
| Prerequisite(s) | The CSMS supports retrieving / deleting CustomerInformation - CustomerCertificate |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| Main (Test scenario) | |
|--|---|
| Charging Station | CSMS |
| 2. The Test System responds with a CustomerInformationResponse with status <i>Accepted</i> | 1. The CSMS sends a CustomerInformationRequest with specific hash data <i><customer certificate hash data></i> . |
| 3. The Test System sends a NotifyCustomerInformationRequest | 4. The CSMS responds with a NotifyCustomerInformationResponse |

| |
|--|
| Tool validations |
| <p>* Step 1:</p> <p>Message CustomerInformationRequest</p> <ul style="list-style-type: none"> - report true - clear true - customerCertificate contains <i><customer certificate hash data></i> |
| <p>Post scenario validations:</p> <ul style="list-style-type: none"> - N/a |

TC_N_34_CSMS: Retrieve Log Information - Rejected

| | |
|-------------------|--|
| Test case name | Retrieve Log Information - Rejected |
| Test case Id | TC_N_34_CSMS |
| Use case Id(s) | N01 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | This test case covers the functionality of getting log information from a Charging Station. The CSMS can request a Charging Station to upload a file with log information to a given location (URL). The format of this log file is not prescribed. The Charging Station successfully uploads a log file and gives information about the status of the upload by sending status notifications to the CSMS. |
| Purpose | To verify if the CSMS is able to request a charging station to successfully upload a log as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a GetLogResponse with status <i>Rejected</i> | 1. The CSMS sends a GetLogRequest |

| |
|-------------------------------------|
| Tool validations |
| N/a |
| Post scenario validations: - N/a |

TC_N_35_CSMS: Retrieve Log Information - Security Log - Success

| | |
|-------------------|--|
| Test case name | Retrieve Log Information - Security Log - Success |
| Test case Id | TC_N_35_CSMS |
| Use case Id(s) | N01 |
| Requirement(s) | |
| System under test | CSMS |
| Description | This test case covers the functionality of getting log information from a Charging Station. The CSMS can request a Charging Station to upload a file with log information to a given location (URL). The format of this log file is not prescribed. The Charging Station successfully uploads a log file and gives information about the status of the upload by sending status notifications to the CSMS. |
| Purpose | To verify if the CSMS is able to request a charging station to successfully upload a log as described at the OCPP specification. |
| Prerequisite(s) | n/a |

Before (Preparations)

Configuration State:

N/a

Memory State:

Charging Station has log information available.

Reusable State:

N/a

Main (Test scenario)

| Charging Station | CSMS |
|---|---|
| 2. The Test System responds with a GetLogResponse with status Accepted | 1. The CSMS sends a GetLogRequest |
| 3. The Test System sends a LogStatusNotificationRequest with - status Uploading - requestId Same Id as the GetLogRequest | 4. The CSMS responds with a LogStatusNotificationResponse . |
| 5. The Test System sends a LogStatusNotificationRequest with - status Uploaded - requestId Same Id as the GetLogRequest | 6. The Test System responds with a LogStatusNotificationResponse . |

Tool validations

* Step 1:

Message **GetLogRequest**- **logType SecurityLog**

Post scenario validations:

- N/a

TC_N_36_CSMS: Retrieve Log Information - Second Request

| | |
|-------------------|--|
| Test case name | Retrieve Log Information - Second Request |
| Test case Id | TC_N_36_CSMS |
| Use case Id(s) | N01 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | This test case covers the functionality of getting log information from a Charging Station. The CSMS can request a Charging Station to upload a file with log information to a given location (URL). The format of this log file is not prescribed. The Charging Station successfully uploads a log file and gives information about the status of the upload by sending status notifications to the CSMS. |
| Purpose | To verify if the CSMS is able to request a second request while the charging station is uploading a log as described at the OCPP specification. |
| Prerequisite(s) | n/a |

Before (Preparations)

Configuration State:

N/a

Memory State:

Charging Station has log information available.

Reusable State:

N/a

Main (Test scenario)

| Charging Station | CSMS |
|--|---|
| 2. The Test System responds with a GetLogResponse with status Accepted | 1. The CSMS sends a GetLogRequest |
| 3. The Test System sends a LogStatusNotificationRequest with - status Uploading - requestId Same Id as the GetLogRequest from Step 1 | 4. The CSMS responds with a LogStatusNotificationResponse . |
| 6. The Test System responds with a GetLogResponse with status AcceptedCanceled | 5. The CSMS sends a GetLogRequest |
| 7. The Test System sends a LogStatusNotificationRequest with - status AcceptedCanceled - requestId Same Id as the GetLogRequest from Step 1 | 8. The CSMS responds with a LogStatusNotificationResponse . |
| 9. The Test System sends a LogStatusNotificationRequest with - status Uploading - requestId Same Id as the GetLogRequest from Step 5 | 10. The CSMS responds with a LogStatusNotificationResponse . |
| 11. The Test System sends a LogStatusNotificationRequest with - status Uploaded - requestId Same Id as the GetLogRequest from Step 5 | 12. The CSMS responds with a LogStatusNotificationResponse . |

Tool validations

N/a

Post scenario validations:

- N/a

TC_N_44_CSMS: Clear Monitoring - Rejected

| | |
|-------------------|--|
| Test case name | Clear Monitoring - Rejected |
| Test case Id | TC_N_44_CSMS |
| Use case Id(s) | N06 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | A monitoring setting can be cleared (removed) by sending a ClearVariableMonitoringRequest with the id of the monitoring setting. |
| Purpose | To verify if the CSMS is able to correctly read the respond from a charging station on a request to clear a monitor that cannot be cleared as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a ClearVariableMonitoringResponse with clearMonitoringResult[0].status Rejected | 1. The CSMS sends a ClearVariableMonitoringRequest |

| |
|-------------------------------------|
| Tool validations |
| N/a |
| Post scenario validations: - N/a |

TC_N_46_CSMS: Clear Customer Information - Update Local Authorization List

| | |
|-------------------|--|
| Test case name | Clear Customer Information - Update Local Authorization List |
| Test case Id | TC_N_46_CSMS |
| Use case Id(s) | N10 |
| Requirement(s) | N10.FR.02, N10.FR.08, D01.FR.01, D01.FR.06, D01.FR.18, |
| System under test | CSMS |
| Description | The CSMS sends a message to the Charging Station to clear (and retrieve) raw customer information, for example to be compliant with local privacy laws. The Charging Station notifies the CSMS by sending one or more reports. |
| Purpose | To verify if the CSMS updates the local authorization list when customer information, which was present in the local authorization list, has been removed as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: A local authorization list with <Configured valid_idtoken_idtoken> is configured. |
| Reusable State: N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual action:</u> Trigger CSMS to CustomerInformationRequest to both report and clear token <Configured valid_idtoken_idtoken> and <Configured valid_idtoken_type> | |
| 2. The Test System responds with a CustomerInformationResponse with status Accepted | 1. The CSMS sends a CustomerInformationRequest |
| 3. The Test System sends a NotifyCustomerInformationRequest | 4. The CSMS responds with a NotifyCustomerInformationResponse . |
| <u>Manual action:</u> If not triggered automatically, trigger CSMS to send SendLocalListRequest with version = <configured local list version> + 1 and updateType = Differential and localAuthorizationList = [{idToken = {<Configured_valid_idtoken>, <Configured valid_idtoken_type>}}] | |
| 6 The Test System responds with a SendLocalListResponse with status Accepted | 5. The CSMS sends a SendLocalListRequest |

| |
|--|
| Tool validations |
| * Step 1: Message CustomerInformationRequest - report true AND - clear true AND - idToken.idToken <Configured valid_idtoken_idtoken> - idToken.type <Configured valid_idtoken_type> * Step 5: Message SendLocalListRequest - updateType Differential - versionNumber <configured local list version> + 1 - localAuthorizationListp[0].idToken contains <configured_valid_idtoken_idtoken> and <configured valid_idtoken_type> - localAuthorizationList[0].idTokenInfo <omitted> |

| |
|--|
| Tool validations |
| Post scenario validations: - All messages have been received |

TC_N_47_CSMS: Get Monitoring report - Report all

| | |
|-------------------|---|
| Test case name | Get Monitoring report - Report all |
| Test case Id | TC_N_47_CSMS |
| Use case Id(s) | N02 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | This test case describes how the CSMS requests the Charging Station to send a report about configured monitoring settings per component and variable. Optionally, this list can be filtered on monitoringCriteria and componentVariables. |
| Purpose | To verify if the CSMS is able to send a get monitor request omitting the monitoringCriteria and componentVariable as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a GetMonitoringReportResponse | 1. The CSMS sends a GetMonitoringReportRequest |
| 3. The Test System sends a NotifyMonitoringReportRequest | 4. The CSMS responds with a NotifyMonitoringReportResponse |
| <u>Note(s):</u> - If tbc is True at Step 3 then step 3 and 4 will be repeated | |

| |
|--|
| Tool validations |
| * Step 1: Message GetMonitoringReportRequest - monitoringCriteria omitted AND - componentVariable omitted. |
| Post scenario validations: - N/a |

TC_N_48_CSMS: Alert Event - Variable monitoring on write only

| | |
|-------------------|---|
| Test case name | Alert Event - Variable monitoring on write only |
| Test case Id | TC_N_48_CSMS |
| Use case Id(s) | N07 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | NotifyEventRequest reports every Component/Variable for which a VariableMonitoring setting was triggered. Only the VariableMonitoring settings that are responsible for triggering an event are included. |
| Purpose | To verify if the CSMS is able to read a request from a trigger from a variablemonitor which is write only as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Test System sends a NotifyEventRequest with eventData.actualValue empty | 2. The CSMS responds with a NotifyEventResponse . |

| |
|-------------------------------------|
| Tool validations |
| N/a |
| Post scenario validations: - N/a |

TC_N_49_CSMS: Alert Event - LowerThreshold/UpperThreshold cleared after reboot

| | |
|-------------------|---|
| Test case name | Alert Event - LowerThreshold/UpperThreshold cleared after reboot |
| Test case Id | TC_N_49_CSMS |
| Use case Id(s) | N07 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | NotifyEventRequest reports every Component/Variable for which a VariableMonitoring setting was triggered. Only the VariableMonitoring settings that are responsible for triggering an event are included. |
| Purpose | To verify if the CSMS is able to read a request when a trigger is cleared after a reboot as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Test System sends a NotifyEventRequest with eventData.cleared <i>true</i> | 2. The CSMS responds with a NotifyEventResponse . |

| |
|-------------------------------------|
| Tool validations |
| N/a |
| Post scenario validations: - N/a |

TC_N_50_CSMS: Alert Event - Periodic Triggered

| | |
|-------------------|---|
| Test case name | Alert Event - Periodic Triggered |
| Test case Id | TC_N_50_CSMS |
| Use case Id(s) | N07 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | NotifyEventRequest reports every Component/Variable for which a VariableMonitoring setting was triggered. Only the VariableMonitoring settings that are responsible for triggering an event are included. |
| Purpose | To verify if the CSMS is able to read a request when a trigger reason is periodic after a reboot as described at the OCPP specification. |
| Prerequisite(s) | n/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Test System sends a NotifyEventRequest with eventData.trigger <i>Periodic</i> | 2. The CSMS responds with a NotifyEventResponse . |

| |
|-------------------------------------|
| Tool validations |
| N/a |
| Post scenario validations: - N/a |

0 Display Message

TC_O_01_CSMS: Set Display Message - Success

| | |
|-------------------|--|
| Test case name | Set Display Message - Success |
| Test case Id | TC_O_01_CSMS |
| Use case Id(s) | O01 |
| Requirement(s) | O01_FR_04 |
| System under test | CSMS |
| Description | This test case describes how the CSMS can be requested to send a SetDisplayMessageRequest to the charging station. Depending on the given parameters the message shall be displayed a certain way and at a certain moment on the Charging Station. These messages are displayed additionally on a Charging Station and are not part of the firmware. |
| Purpose | To verify if the CSMS is able to send the request according to the DisplayMessage mechanism as described in the OCPP specification. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action:</u> Request the CSMS to send a SetDisplayMessageRequest. | |
| 2. The Test System responds with a SetDisplayMessageResponse with: status Accepted | 1. The CSMS sends a SetDisplayMessageRequest with: state <Configured display message state> |

| |
|--|
| Tool validations |
| <p>* Step 1:</p> <p>Message SetDisplayMessageRequest</p> <ul style="list-style-type: none"> - message.id <Generated Id> - message.priority <Configured Priority> - message.message.format <Configured Format> - message.state <Configured State> |
| <p>Post scenario validations:</p> <ul style="list-style-type: none"> - N/a |

TC_O_02_CSMS: Get all Display Messages - Success

| | |
|-------------------|---|
| Test case name | Get all Display Messages - Success |
| Test case Id | TC_O_02_CSMS |
| Use case Id(s) | O03 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | This test case describes how a CSO can request all the installed DisplayMessages configured via OCPP in a Charging Station. The Charging Station can remove messages when they are out-dated, or transactions have ended. It can be very useful for a CSO to be able to view to current list of messages, so the CSO knows which messages are (still) configured. |
| Purpose | To verify if the CSMS is able to send the request to get the DisplayMessages according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | N/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: A display message is configured. |
| Reusable State: N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a GetDisplayMessagesResponse with status Accepted | 1. The CSMS sends a GetDisplayMessagesRequest |
| 3. The Test System sends a NotifyDisplayMessagesRequest | 4. The CSMS responds with a NotifyDisplayMessagesResponse |

| |
|--|
| Tool validations |
| * Step 1: Message GetDisplayMessagesRequest - requestId <Generated Id> - id <Omitted> - priority <Omitted> - state <Omitted> |
| Post scenario validations: - N/a |

TC_O_03_CSMS: Get all Display Messages - No DisplayMessages configured

| | |
|-------------------|--|
| Test case name | Get all Display Messages - No DisplayMessages configured |
| Test case Id | TC_O_03_CSMS |
| Use case Id(s) | O03 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | This test case describes how a CSO can request all the installed DisplayMessages configured via OCPP in a Charging Station. The Charging Station can remove messages when they are out-dated, or transactions have ended. It can be very useful for a CSO to be able to view the current list of messages, so the CSO knows which messages are (still) configured. |
| Purpose | To verify if the CSMS can request to get all display messages according to the DisplayMessage mechanism as described in the OCPP specification when no messages are configured. |
| Prerequisite(s) | N/a |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State:

N/a

Main (Test scenario)

| Charging Station | CSMS |
|---|--|
| 2. The Test System responds with a GetDisplayMessagesResponse with status Unknown | 1. The CSMS sends a GetDisplayMessagesRequest |

Tool validations

* Step 1:

Message **GetDisplayMessagesRequest**- **requestId** <Generated request id>

Post scenario validations:

- N/a

TC_O_04_CSMS: Clear Display Message - Success

| | |
|-------------------|---|
| Test case name | Clear Display Message - Success |
| Test case Id | TC_O_04_CSMS |
| Use case Id(s) | O05 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | This test case describes how a CSO can remove a specific message, configured via OCPP in a Charging Station. |
| Purpose | To verify if the CSMS is able to request the Charging Station to clear a message according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | N/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: A display message is configured. |
| Reusable State: N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <i>Note: As a help method, a <code>GetDisplayMessagesRequest</code> is requested first for CSMS's that implemented their <code>ClearDisplayMessage</code> as a combined feature.</i> | |
| 2. The Test System responds with a ClearDisplayMessageResponse with status <i>Accepted</i> | 1. The CSMS sends a ClearDisplayMessageRequest |

| |
|---|
| Tool validations |
| * Step 1: Message ClearDisplayMessageRequest - id <Generated Id from set display message> |
| Post scenario validations: - N/a |

TC_O_05_CSMS: Clear Display Message - Unknown Key

| | |
|--------------------------|---|
| Test case name | Clear Display Message - Unknown Key |
| Test case Id | TC_O_05_CSMS |
| Use case Id(s) | O05 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | This test case describes how a CSO can remove a specific message, configured via OCPP in a Charging Station. |
| Purpose | To verify if the CSMS is able to request the Charging Station to clear a message according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | If the CSMS supports sending a ClearDisplayMessageRequest with an unknown id. |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a ClearDisplayMessageResponse with status <i>Unknown</i> | 1. The CSMS sends a ClearDisplayMessageRequest |

| |
|--|
| Tool validations |
| N/a |
| Post scenario validations: - N/a |

TC_O_06_CSMS: Set Display Message - Specific transaction - Success

| | |
|-------------------|--|
| Test case name | Set Display Message - Specific transaction - Success |
| Test case Id | TC_O_06_CSMS |
| Use case Id(s) | O02 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | This test case describes how a CSO can set a message to be displayed on a Charging Station for a specific transaction. Depending on the given parameters the message shall be displayed a certain way on the Charging Station. |
| Purpose | To verify if the CSMS is able to send a display message correctly according the mechanism as described in the OCPP specification for a specific transaction. |
| Prerequisite(s) | N/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: State is <i>EnergyTransferStarted</i> |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action</u> : Request the CSMS to send a display message for a specific transaction. | |
| 2. The Test System responds with a SetDisplayMessageResponse with status <i>Accepted</i> | 1. The CSMS sends a SetDisplayMessageRequest |
| 3. Execute Reusable State <i>EVDisconnected</i> | |

| |
|--|
| Tool validations |
| * Step 1: Message SetDisplayMessageRequest - message.transactionId Same ID as previously returned by the Charging Station AND - message.priority <Configured Priority> |
| Post scenario validations: - N/a |

TC_O_07_CSMS: Get a Specific Display Message - Id

| | |
|--------------------------|--|
| Test case name | Get a Specific Display Message - Id |
| Test case Id | TC_O_07_CSMS |
| Use case Id(s) | O04 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | This test case describes how a CSO can request specific installed DisplayMessages configured via OCPP in a Charging Station. The Charging Station can remove messages when they are out-dated, or transactions have ended. It can be very useful for a CSO to be able to view to current list of messages, so the CSO knows which messages are (still) configured. |
| Purpose | To verify if the CSMS is able to request a specific id message from the charging station according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | N/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: A display message is configured. |
| Reusable State: N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a GetDisplayMessagesResponse with status Accepted | 1. The CSMS sends a GetDisplayMessagesRequest |
| 3. The Test System sends a NotifyDisplayMessagesRequest | 4. The CSMS responds with a NotifyDisplayMessagesResponse |

| |
|--|
| Tool validations |
| <p>* Step 1:</p> <p>Message GetDisplayMessagesRequest</p> <ul style="list-style-type: none"> - id <Configured_Id> - priority <Omitted> - state <Omitted> - requestId <Generated Id> |
| <p>Post scenario validations:</p> <ul style="list-style-type: none"> - N/a |

TC_O_08_CSMS: Get a Specific Display Message - Priority

| | |
|-------------------|--|
| Test case name | Get a Specific Display Message - Priority |
| Test case Id | TC_O_08_CSMS |
| Use case Id(s) | O04 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | This test case describes how a CSO can request specific installed DisplayMessages configured via OCPP in a Charging Station. The Charging Station can remove messages when they are out-dated, or transactions have ended. It can be very useful for a CSO to be able to view to current list of messages, so the CSO knows which messages are (still) configured. |
| Purpose | To verify if the CSMS is able to request specific priority messages from the charging station according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | N/a |

Before (Preparations)

Configuration State:

N/a

Memory State:

A message with <Configured_Priority> is configured

Reusable State:

N/a

Main (Test scenario)

| Charging Station | CSMS |
|--|--|
| 2. The Test System responds with a GetDisplayMessagesResponse with status Accepted | 1. The CSMS sends a GetDisplayMessagesRequest |
| 3. The Test System sends a NotifyDisplayMessagesRequest | 4. The CSMS responds with a NotifyDisplayMessagesResponse |

Tool validations

* Step 1:

Message **GetDisplayMessagesRequest**

- **priority** <Configured_Priority>
- **id** <Omitted>
- **state** <Omitted>
- **requestId** <Generated Id>

Post scenario validations:

- N/a

TC_O_09_CSMS: Get a Specific Display Message - State

| | |
|-------------------|--|
| Test case name | Get a Specific Display Message - State |
| Test case Id | TC_O_09_CSMS |
| Use case Id(s) | 004 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | This test case describes how a CSO can request specific installed DisplayMessages configured via OCPP in a Charging Station. The Charging Station can remove messages when they are out-dated, or transactions have ended. It can be very useful for a CSO to be able to view to current list of messages, so the CSO knows which messages are (still) configured. |
| Purpose | To verify if the CSMS is able to request specific state messages from the charging station according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | N/a |

| |
|---|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: A message with <Configured_State> is configured |
| Reusable State: N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a GetDisplayMessagesResponse with status <i>Accepted</i> | 1. The CSMS sends a GetDisplayMessagesRequest |
| 3. The Test System sends a NotifyDisplayMessagesRequest | 4. The CSMS responds with a NotifyDisplayMessagesResponse . |

| |
|---|
| Tool validations |
| * Step 1: Message GetDisplayMessagesRequest - state <Configured_State> - priority <Omitted> - id <Omitted> - requestId <Generated Id> |
| Post scenario validations: - N/a |

TC_O_10_CSMS: Set Display Message - Specific transaction - UnknownTransaction

| | |
|-------------------|---|
| Test case name | Set Display Message - Specific transaction - UnknownTransaction |
| Test case Id | TC_O_10_CSMS |
| Use case Id(s) | O02 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | This test case describes how a CSMS can attempt to set a DisplayMessage for a transactionId that the CS does not know. The CS will respond with a SetDisplayMessageResponse status of UnknownTransaction. |
| Purpose | To verify if the CSMS is able to send a display message correctly according the mechanism as described in the OCPP specification for a specific transaction. |
| Prerequisite(s) | If the CSMS supports sending a SetDisplayMessageRequest with a transactionId for a transaction that does not exist. |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: State is <i>EnergyTransferStarted</i> |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action</u> : Request the CSMS to send a display message for a specific transaction. | |
| 2. The Test System responds with a SetDisplayMessageResponse with status <i>UnknownTransaction</i> | 1. The CSMS sends a SetDisplayMessageRequest |

| |
|--|
| Tool validations |
| * Step 1: Message SetDisplayMessageRequest - message.transactionId not omit AND - message.priority <Configured Priority> |
| Post scenario validations: - N/a |

TC_O_11_CSMS: Get a Specific Display Message - Unknown parameters

| | |
|-------------------|--|
| Test case name | Get a Specific Display Message - Unknown parameters |
| Test case Id | TC_O_11_CSMS |
| Use case Id(s) | 004 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | This test case describes how a CSO can request specific installed DisplayMessages configured via OCPP in a Charging Station. The Charging Station can remove messages when they are out-dated, or transactions have ended. It can be very useful for a CSO to be able to view to current list of messages, so the CSO knows which messages are (still) configured. |
| Purpose | To verify if the CSMS is able to request a specific id message from the charging station according to the mechanism as described in the OCPP specification. |
| Prerequisite(s) | If the CSMS is able to send a GetDisplayMessage with an unknown id. |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: A display message is configured. |
| Reusable State: N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a GetDisplayMessagesResponse with status <i>Unknown</i> | 1. The CSMS sends a GetDisplayMessagesRequest |

| |
|--|
| Tool validations |
| * Step 1: Message GetDisplayMessagesRequest - id <A different generated Id> - requestId <Generated Id> |
| Post scenario validations: - N/a |

TC_O_12_CSMS: Set Display Message - Replace DisplayMessage

| | |
|-------------------|---|
| Test case name | Set Display Message - Replace DisplayMessage |
| Test case Id | TC_O_12_CSMS |
| Use case Id(s) | O06 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | This test case describes how a CSO can replace a DisplayMessage that is previously configured in a Charging Station. Replace the message content, but also all the given parameters with the new one. |
| Purpose | To verify if the CSMS is able to request to replace a display message according to the DisplayMessage mechanism as described in the OCPP specification. |
| Prerequisite(s) | N/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: A display message is configured. |
| Reusable State: N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action:</u> - Request the CSMS to sent a display message with the same id as already configured one | |
| 2. The Test System responds with a SetDisplayMessageResponse with status <i>Accepted</i> | 1. The CSMS sends a SetDisplayMessageRequest with: message.id <Configured_Id> message.priority <Configured Priority> |

| |
|--|
| Tool validations |
| * Step 2: Message SetDisplayMessageRequest - message.id <Configured_Id> - message.priority <Configured Priority> |
| Post scenario validations: - N/a |

TC_O_13_CSMS: Set Display Message - Display message at StartTime

| | |
|-------------------|--|
| Test case name | Set Display Message - Display message at StartTime |
| Test case Id | TC_O_13_CSMS |
| Use case Id(s) | 001 |
| Requirement(s) | 001_FR_05 |
| System under test | CSMS |
| Description | This test case describes how the CSMS can be requested to send a SetDisplayMessageRequest to the charging station. Depending on the given parameters the message shall be displayed a certain way and at a certain moment on the Charging Station. These messages are displayed additionally on a Charging Station and are not part of the firmware. |
| Purpose | To verify if the CSMS is able to send the request with a startTime according to the DisplayMessage mechanism as described in the OCPP specification. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action</u> : Request the CSMS to send a SetDisplayMessageRequest with a startTime. | |
| 2. The Test System responds with a SetDisplayMessageResponse with status Accepted | 1. The CSMS sends a SetDisplayMessageRequest |

| |
|--|
| Tool validations |
| * Step 1: Message SetDisplayMessageRequest - message.id <Generated Id> - message.startDateTime <Configured startDateTime> |
| Post scenario validations: - N/a |

TC_O_14_CSMS: Set Display Message - Remove message after EndTime

| | |
|-------------------|--|
| Test case name | Set Display Message - Remove message after EndTime |
| Test case Id | TC_O_14_CSMS |
| Use case Id(s) | O01 |
| Requirement(s) | O01_FR_05 |
| System under test | CSMS |
| Description | This test case describes how the CSMS can be requested to send a SetDisplayMessageRequest to the charging station. Depending on the given parameters the message shall be displayed a certain way and at a certain moment on the Charging Station. These messages are displayed additionally on a Charging Station and are not part of the firmware. |
| Purpose | To verify if the CSMS is able to send the request with a endTime according to the DisplayMessage mechanism as described in the OCPP specification. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action:</u> Request the CSMS to send a SetDisplayMessageRequest with a endTime. | |
| 2. The Test System responds with a SetDisplayMessageResponse with status Accepted | 1. The CSMS sends a SetDisplayMessageRequest |

| |
|--|
| Tool validations |
| * Step 1: Message SetDisplayMessageRequest - message.id <Generated Id> - message.endTime <Configured endTime> |
| Post scenario validations: - N/a |

TC_O_17_CSMS: Set Display Message - NotSupportedPriority

| | |
|-------------------|--|
| Test case name | Set Display Message - NotSupportedPriority |
| Test case Id | TC_O_17_CSMS |
| Use case Id(s) | 001 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | This test case describes how the CSMS can be requested to send a SetDisplayMessageRequest to the charging station. Depending on the given parameters the message shall be displayed a certain way and at a certain moment on the Charging Station. These messages are displayed additionally on a Charging Station and are not part of the firmware. |
| Purpose | To verify if the CSMS is able to send a display message with a specific priority, on which the Charging station responds not supported, according to the DisplayMessage mechanism as described in the OCPP specification. |
| Prerequisite(s) | N/a |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State:

N/a

Main (Test scenario)

| Charging Station | CSMS |
|---|--|
| 2. The Test System responds with a SetDisplayMessageResponse with status NotSupportedPriority | 1. The CSMS sends a SetDisplayMessageRequest |

Tool validations

* Step 1:

Message SetDisplayMessageRequest

- message.id <Generated Id>

- message.priority <Configured priority>

Post scenario validations:

- N/a

TC_O_18_CSMS: Set Display Message - NotSupportedState

| | |
|-------------------|--|
| Test case name | Set Display Message - NotSupportedState |
| Test case Id | TC_O_18_CSMS |
| Use case Id(s) | 001 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | This test case describes how the CSMS can be requested to send a SetDisplayMessageRequest to the charging station. Depending on the given parameters the message shall be displayed a certain way and at a certain moment on the Charging Station. These messages are displayed additionally on a Charging Station and are not part of the firmware. |
| Purpose | To verify if the CSMS is able to send a display message with a specific state, on which the Charging station responds not supported, according to the DisplayMessage mechanism as described in the OCPP specification. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a SetDisplayMessageResponse with status <i>NotSupportedState</i> | 1. The CSMS sends a SetDisplayMessageRequest |

| |
|--|
| Tool validations |
| * Step 1: Message SetDisplayMessageRequest - message.id <Generated Id> - message.state <Configured state> |
| Post scenario validations: - N/a |

TC_O_19_CSMS: Set Display Message - NotSupportedMessageFormat

| | |
|-------------------|---|
| Test case name | Set Display Message - NotSupportedMessageFormat |
| Test case Id | TC_O_19_CSMS |
| Use case Id(s) | 001 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | This test case describes how the CSMS can be requested to sent an SetDisplayMessageRequest to the charging station. Depending on the given parameters the message shall be displayed a certain way and at a certain moment on the Charging Station. These messages are displayed additionally on a Charging Station and are not part of the firmware. |
| Purpose | To verify if the CSMS is able to send a display message with a specific MessageFormat, on which the Charging station responds not supported, according to the DisplayMessage mechanism as described in the OCPP specification. |
| Prerequisite(s) | N/a |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State:

N/a

Main (Test scenario)

| Charging Station | CSMS |
|--|--|
| 2. The Test System responds with a SetDisplayMessageResponse with status NotSupportedMessageFormat | 1. The CSMS sends a SetDisplayMessageRequest |

Tool validations

* Step 1:

Message SetDisplayMessageRequest
- message.id <Generated Id>

Post scenario validations:

- N/a

TC_O_25_CSMS: Set Display Message - Send Specific state

| | |
|-------------------|---|
| Test case name | Set Display Message - Send Specific state |
| Test case Id | TC_O_25_CSMS |
| Use case Id(s) | 001 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | This test case describes how the CSMS can be requested to sent an SetDisplayMessageRequest to the charging station. Depending on the given parameters the message shall be displayed a certain way and at a certain moment on the Charging Station. These messages are displayed additionally on a Charging Station and are not part of the firmware. |
| Purpose | To verify if the CSMS is able to send a display messages with a "Charging" state according to the DisplayMessage mechanism as described in the OCPP specification. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a SetDisplayMessageResponse with status <i>Accepted</i> | 1. The CSMS sends a SetDisplayMessageRequest |

| |
|---|
| Tool validations |
| * Step 1: Message SetDisplayMessageRequest - message.id <Configured_Id> - message.state <Configured State> |
| Post scenario validations: - N/a |

TC_O_26_CSMS: Set Display Message - Rejected

| | |
|-------------------|--|
| Test case name | Set Display Message - Rejected |
| Test case Id | TC_O_26_CSMS |
| Use case Id(s) | O01 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | This test case describes how the CSMS can be requested to send a SetDisplayMessageRequest to the charging station. Depending on the given parameters the message shall be displayed a certain way and at a certain moment on the Charging Station. These messages are displayed additionally on a Charging Station and are not part of the firmware. |
| Purpose | To verify if the CSMS is able to send the request according to the DisplayMessage mechanism as described in the OCPP specification which gets rejected. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action:</u> Request the CSMS to send a <code>SetDisplayMessageRequest</code> with a Normal Cycle priority. | |
| 2. The Test System responds with a <code>SetDisplayMessageResponse</code> with <code>status Rejected</code> | 1. The CSMS sends a <code>SetDisplayMessageRequest</code> |

| |
|--|
| Tool validations |
| * Step 1: Message SetDisplayMessageRequest - message.id <Generated Id> - message.priority <Configured Priority> |
| Post scenario validations: - N/a |

TC_O_27_CSMS: Set Display Message - Specific transaction - Display message at StartTime

| | |
|-------------------|--|
| Test case name | Set Display Message - Specific transaction - Display message at StartTime |
| Test case Id | TC_O_27_CSMS |
| Use case Id(s) | O02 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | This test case describes how the CSMS can be requested to send a SetDisplayMessageRequest to the charging station. Depending on the given parameters the message shall be displayed a certain way and at a certain moment on the Charging Station. These messages are displayed additionally on a Charging Station and are not part of the firmware. |
| Purpose | To verify if the CSMS is able to send the request with a startTime for a specific transaction according to the DisplayMessage mechanism as described in the OCPP specification. |
| Prerequisite(s) | N/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a1 |
| Reusable State: State is <i>EnergyTransferStarted</i> |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 2. The Test System responds with a SetDisplayMessageResponse with <i>status Accepted</i> | 1. The CSMS sends a SetDisplayMessageRequest |

| |
|--|
| Tool validations |
| <p>* Step 1:</p> <p>Message SetDisplayMessageRequest</p> <ul style="list-style-type: none"> - message.state is <omitted> - message.startDateTime is <Configured startDateTime> - message.endDateTime is <omitted> - message.transactionId is <Generated transactionId from Before> |
| <p>Post scenario validations:</p> <ul style="list-style-type: none"> - N/a |

TC_O_28_CSMS: Set Display Message - Specific transaction - Remove message after EndTime

| | |
|-------------------|--|
| Test case name | Set Display Message - Specific transaction - Remove message after EndTime |
| Test case Id | TC_O_28_CSMS |
| Use case Id(s) | O02 |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | This test case describes how the CSMS can be requested to send a SetDisplayMessageRequest to the charging station. Depending on the given parameters the message shall be displayed a certain way and at a certain moment on the Charging Station. These messages are displayed additionally on a Charging Station and are not part of the firmware. |
| Purpose | To verify if the CSMS is able to send the request with an endTime for a specific transaction according to the DisplayMessage mechanism as described in the OCPP specification. |
| Prerequisite(s) | N/a |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State: State is <i>EnergyTransferStarted</i> |

| Main (Test scenario) | |
|--|---|
| Charging Station | CSMS |
| 2. The Test System responds with a SetDisplayMessageResponse with status <i>Accepted</i> | 1. The CSMS sends a SetDisplayMessageRequest |

| |
|--|
| Tool validations |
| <p>* Step 1:</p> <p>Message SetDisplayMessageRequest</p> <ul style="list-style-type: none"> - message.state is <omitted> - message.startDateTime is <omitted> - message.endDateTime is <Configured endDateTime> - message.transactionId is <Generated transactionId from Before> |
| <p>Post scenario validations:</p> <ul style="list-style-type: none"> - N/a |

P Data Transfer

TC_P_02_CSMS: Data Transfer to the CSMS - Rejected / Unknown VendorId / Unknown MessageId

| | |
|-------------------|--|
| Test case name | Data Transfer to the CSMS - Rejected / Unknown VendorId / Unknown MessageId |
| Test case Id | TC_P_02_CSMS |
| Use case Id(s) | P02 |
| Requirement(s) | P02.FR.06, P02.FR.07 |
| System under test | CSMS |
| Description | The DataTransfer message to send information for functions that are not supported by OCPP. |
| Purpose | To verify whether the CSMS is able to handle receiving a DataTransferRequest, even if it does not support any vendor-specific implementations. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Test System sends a DataTransferRequest with vendorId <Configured vendorId> messageId <Configured messageId> | 2. The CSMS responds with a DataTransferResponse |

| |
|--|
| Tool validations |
| * Step 2: Message: DataTransferResponse - status must be <i>UnknownVendorId</i> OR <i>UnknownMessageId</i> OR <i>Rejected</i> (Rejected will also be allowed, because there are implementers that like to just reject the message when the Charging Station does not support any vendor-specific features. |
| Post scenario validations: N/a |

TC_P_03_CSMS: CustomData - Receive custom data

| | |
|-------------------|---|
| Test case name | CustomData - Receive custom data |
| Test case Id | TC_P_03_CSMS |
| Use case Id(s) | N/a |
| Requirement(s) | N/a |
| System under test | CSMS |
| Description | Checks if the CSMS is able to receive custom data. |
| Purpose | To verify whether the CSMS is able to handle receiving custom data. |
| Prerequisite(s) | N/a |

| |
|-----------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| Main (Test scenario) | |
|---|---|
| Charging Station | CSMS |
| 1. The Test System sends a StatusNotificationRequest with customData <customData> | 2. The CSMS responds with a StatusNotificationResponse |
| 3. The Test System sends a TransactionEventRequest with customData customData transactionInfo.customData <customData> | 4. The CSMS responds with a TransactionEventResponse |

| |
|-----------------------------------|
| Tool validations |
| N/a |
| Post scenario validations: N/a |

Reusable states

Testcases can refer to a reusable state at the before or main stage. The steps described at the reusable state will be executed and then it will return to the testcase that called the reusable state.

Booted

| | |
|--------------------------|---|
| State | Booted |
| System under test | CSMS |
| Description | This state will simulate that the Charging Station is completely power cycled. The Test System end in a state where it is "booted" back up and is in idle mode. |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Test System sends a BootNotificationRequest with reason <i>PowerUp</i> chargingStation.model <Configured model> chargingStation.vendorName <Configured vendorName> | 2. The CSMS responds with a BootNotificationResponse |
| 3. The Test System notifies the CSMS about the current state of all connectors. Message: StatusNotificationRequest with connectorStatus <i>Available</i> Message: NotifyEventRequest with trigger <i>Delta</i> actualValue <i>"Available"</i> component.name <i>"Connector"</i> variable.name <i>"AvailabilityState"</i> | 4. The CSMS responds accordingly. |

| |
|--|
| Tool validations |
| * Step 2: Message: BootNotificationResponse - status <i>Accepted</i> |
| Post scenario validations: State is <i>Booted</i> |

Reserved

| | |
|--------------------------|--|
| State | Reserved |
| System under test | CSMS |
| Description | This state will simulate a reservation for a specified evse. |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| Manual Action: <i>Trigger the CSMS to send a ReserveNowRequest for specific EVSE.</i> | |
| 2. The Test System responds with a ReserveNowResponse With status <i>Accepted</i> | 1. The CSMS sends a ReserveNowRequest |
| 3. The Test System notifies the CSMS about the current state of the connector(s) of the Specified EVSE Message: StatusNotificationRequest with connectorStatus <i>Reserved</i> Message: NotifyEventRequest with trigger <i>Delta</i> actualValue <i>"Reserved"</i> component.name <i>"Connector"</i> variable.name <i>"AvailabilityState"</i> | 4. The CSMS responds accordingly. |

| |
|---|
| Tool validations |
| * Step 1: Message: ReserveNowRequest - evseld must be <i><Specified evseld></i> - connectorType must be omitted - idToken.idToken <i><Configured valid_idtoken_idtoken></i> - idToken.type <i><Configured valid_idtoken_type></i> |
| Post scenario validations: State is <i>Reserved</i> |

Unavailable

| | |
|--------------------------|---|
| State | Unavailable |
| System under test | CSMS |
| Description | This state will simulate that Charging Station / EVSEs / connectors are set to AvailabilityState Unavailable. |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|--|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| Manual Action: Request the CSMS to change the availability of the specified components to Inoperative. | |
| 2. The Test System responds with a ChangeAvailabilityResponse with status <i>Accepted</i> | 1. The CSMS sends a ChangeAvailabilityRequest |
| 3. The Test System notifies the CSMS about the current state of all connectors belonging to the specified EVSE (and optionally also from the EVSE itself). Message: StatusNotificationRequest - connectorStatus <i>Unavailable</i> Message: NotifyEventRequest - trigger <i>Delta</i> - actualValue <i>"Unavailable"</i> - component.name <i>"ChargingStation" / EVSE / Connector</i> - variable.name <i>"AvailabilityState"</i> | 4. The CSMS responds accordingly. |

| |
|--|
| Tool validations |
| * Step 1: Message ChangeAvailabilityRequest - operationalStatus <i>Inoperative</i> - evse <i><Specified evseld></i> - connectorId <i>omitted</i> |
| Post scenario validations: State is <i>Unavailable</i> |

EVConnectedPreSession

| | |
|--------------------------|--|
| State | EVConnectedPreSession |
| System under test | CSMS |
| Description | This state will simulate that the EV and EVSE of the simulated Charging Station are connected. |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State(s):

N/a

Main (Test scenario)

| Charging Station | CSMS |
|--|---|
| 1. The Test System notifies the CSMS about the status change of the connector Message: StatusNotificationRequest - connectorStatus is <i>Occupied</i> Message: NotifyEventRequest - trigger is <i>Delta</i> - actualValue is <i>Occupied</i> - component.name is <i>Connector</i> - variable.name is <i>AvailabilityState</i> | 2. The CSMS responds accordingly. |
| 3. The Test System sends a TransactionEventRequest With triggerReason is <i>CablePluggedIn</i> transactionInfo.chargingState is <i>EVConnected</i> evse.id <Configured evseld> evse.connectorId <Configured connectorId> If State is <i>Authorized</i> then eventType is <i>Updated</i> else eventType is <i>Started</i> | 4. The CSMS responds with a TransactionEventResponse |

Tool validations

N/a

Post scenario validations:

State is *EVConnectedPreSession*

Authorized

| | |
|--------------------------|--|
| State | Authorized |
| System under test | CSMS |
| Description | This state will simulate that the EV Driver is locally authorizing to start a transaction on the simulated Charging Station. |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Test System sends an AuthorizeRequest With idToken.idToken <Configured valid_idtoken_idtoken> idToken.type <Configured valid_idtoken_type> | 2. The CSMS responds with an AuthorizeResponse |
| 3. The Test System sends a TransactionEventRequest With triggerReason is <i>Authorized</i> idToken.idToken <Configured valid_idtoken_idtoken> idToken.type <Configured valid_idtoken_type> If State is <i>EVConnectedPreSession</i> then eventType is <i>Updated</i> else eventType is <i>Started</i> | 4. The CSMS responds with a TransactionEventResponse |

| |
|---|
| Tool validations |
| * Step 2: Message: AuthorizeResponse - idTokenInfo.status must be <i>Accepted</i> * Step 4: Message: TransactionEventResponse - idTokenInfo.status must be <i>Accepted</i> |
| Post scenario validations: State is <i>Authorized</i> |

EnergyTransferStarted

| | |
|--------------------------|---|
| State | EnergyTransferStarted |
| System under test | CSMS |
| Description | This state will simulate that there is transferring energy between the EV and EVSE of the simulated Charging Station. |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State(s):

If **State** is NOT *Authorized* then execute **Reusable State** *Authorized*

If **EVConnected** is *true*, then proceed to part 2

Else proceed to part 1.

Main (Part 1) (Test scenario)

| Charging Station | CSMS |
|---|--|
| 1. The Test System notifies the CSMS about the status change of the connector. Message: StatusNotificationRequest - connectorStatus is <i>Occupied</i> Message: NotifyEventRequest - trigger is <i>Delta</i> - actualValue is <i>Occupied</i> - component.name is <i>Connector</i> - variable.name is <i>AvailabilityState</i> | 2. The CSMS responds accordingly. |
| 3. The Test System sends a TransactionEventRequest With triggerReason is <i>CablePluggedIn</i> transactionInfo.chargingState is <i>EVConnected</i> evse.id <Configured evseld> evse.connectorId <Configured connectorId> eventType is <i>Updated</i> | 4. The CSMS responds with a TransactionEventResponse |

Tool validations

N/a

Main (Part 2) (Test scenario)

| Charging Station | CSMS |
|---|--|
| 5. The Test System sends a TransactionEventRequest With triggerReason is <i>ChargingStateChanged</i> transactionInfo.chargingState is <i>Charging</i> eventType is <i>Updated</i> | 6. The CSMS responds with a TransactionEventResponse |

Tool validations

N/a

Post scenario validations:

State is *EnergyTransferStarted*

EVConnected is *true*

EnergyTransferSuspended

| | |
|--------------------------|--|
| State | EnergyTransferSuspended |
| System under test | CSMS |
| Description | This state will simulate that the Charging Station is in a state where the energy transfer is suspended by the EV. |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): If State is NOT <i>EnergyTransferStarted</i> then execute Reusable State <i>EnergyTransferStarted</i> |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| Notes(s): The tool will wait for <Configured Transaction Duration> seconds | |
| 1. The Test System sends a TransactionEventRequest With triggerReason is <i>ChargingStateChanged</i> transactionInfo.chargingState is <i>SuspendedEV</i> | 2. The CSMS responds with a TransactionEventResponse |

| |
|---|
| Tool validations |
| N/a |
| Post scenario validations: State is <i>EnergyTransferSuspended</i> |

StopAuthorized

| | |
|--------------------------|--|
| State | StopAuthorized |
| System under test | CSMS |
| Description | This state will simulate that the Charging Station is in a state where the charging session is authorized to stop. |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): If State is NOT <i>EnergyTransferStarted</i> then execute Reusable State <i>EnergyTransferStarted</i> |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| Notes(s): The tool will wait for <Configured Transaction Duration> seconds | |
| 1. The Test System sends a TransactionEventRequest With triggerReason is <i>StopAuthorized</i> eventType is <i>Updated</i> | 2. The CSMS responds with a TransactionEventResponse |

| |
|--|
| Tool validations |
| * Step 2: Message: TransactionEventResponse - idTokenInfo.status must be <i>Accepted</i> |
| Post scenario validations: State is <i>StopAuthorized</i> |

EVConnectedPostSession

| | |
|--------------------------|---|
| State | EVConnectedPostSession |
| System under test | CSMS |
| Description | This state will simulate that the Charging Station is in a state where the energy transfer has been stopped and the transaction is NOT authorized to resume energy transfer without re-authorization. |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): If State is NOT <i>StopAuthorized</i> then execute Reusable State <i>StopAuthorized</i> |

| | |
|---|---|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Test System sends a TransactionEventRequest With triggerReason is <i>ChargingStateChanged</i> transactionInfo.chargingState is <i>EVConnected</i> eventType is <i>Updated</i> | 2. The CSMS responds with a TransactionEventResponse |

| |
|--|
| Tool validations |
| N/a |
| Post scenario validations: State is <i>EVConnectedPostSession</i> |

EVDisconnected

| | |
|--------------------------|---|
| State | EVDisconnected |
| System under test | CSMS |
| Description | This state will simulate that the EV and EVSE of the simulated Charging Station are disconnected, after the charging session is authorized to stop. |

| |
|--|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): If State is NOT <i>EVConnectedPostSession</i> then execute Reusable State <i>EVConnectedPostSession</i> |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| 1. The Test System notifies the CSMS about the status change of the connector. Message: StatusNotificationRequest - connectorStatus is <i>Available</i> Message: NotifyEventRequest - trigger is <i>Delta</i> - actualValue is <i>Available</i> - component.name is <i>Connector</i> - variable.name is <i>AvailabilityState</i> | 2. The CSMS responds accordingly. |
| 3. The Test System sends a TransactionEventRequest With triggerReason is <i>EVCommunicationLost</i> transactionInfo.chargingState is <i>Idle</i> transactionInfo.stoppedReason is <i>EVDisconnected</i> eventType is <i>Ended</i> | 4. The CSMS responds with a TransactionEventResponse |

| |
|--|
| Tool validations |
| N/a |
| Post scenario validations: State is <i>EVDisconnected</i> |

GetInstalledCertificates

| | |
|--------------------------|--|
| State | GetInstalledCertificates |
| System under test | CSMS |
| Description | The hashData from installed certificates of the specified type will be retrieved from the Charging Station |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State(s):

N/a

Main (Test scenario)

| | |
|--|---|
| Charging Station | CSMS |
| Manual Action: <i>Trigger the CSMS to send a GetInstalledCertificateIdsRequest with certificateType _<Specified certificateType></i> | |
| 2. The Test System responds with a GetInstalledCertificateIdsResponse With status is <i>Accepted</i> certificateHashDataChain contains an entry with following values: certificateHashDataChain[0].certificateType is <i><Specified certificateType></i> certificateHashDataChain[0].certificateHashData contains <i><HashData from the configured certificate of the specified certificateType></i> | 1. The CSMS sends a GetInstalledCertificateIdsRequest |

Tool validations

* Step 1:

Message: **GetInstalledCertificateIdsRequest**- **certificateType** must be *<Specified certificateType>*

Post scenario validations:

Certificate of the specified certificateType is retrieved from the Charging Station.

CertificateInstalled

| | |
|-------------------|--|
| State | CertificateInstalled |
| System under test | CSMS |
| Description | A pre configured certificate of the specified certificateType will be installed. |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State(s):

N/a

Main (Test scenario)

| | |
|---|---|
| Charging Station | CSMS |
| Manual Action: Trigger the CSMS to send an InstallCertificateRequest with certificateType <Specified certificateType> | |
| 2. The Test System responds with a InstallCertificateResponse With status is Accepted | 1. The CSMS sends a InstallCertificateRequest |

Tool validations

* Step 1:

Message: InstallCertificateRequest

- certificateType must be <Specified certificateType>

- certificate must be <The configured certificate of the specified certificateType.>

Post scenario validations:

Certificate of the specified certificateType is stored at the Charging Station.

ISO15118SmartCharging

| | |
|-------------------|-----------------------|
| State | ISO15118SmartCharging |
| System under test | CSMS |
| Description | |

Before (Preparations)

Configuration State:

N/a

Memory State:

N/a

Reusable State(s):

N/a

Main (Test scenario)

| | |
|--|---|
| Charging Station | CSMS |
| 1. The Test System sends a NotifyEVChargingNeedsRequest with evseld <Configured evseld> maxScheduleTuples & chargingNeeds <Configured values from mock EV> | 2. The CSMS responds with a NotifyEVChargingNeedsResponse. |

| Main (Test scenario) | |
|--|--|
| 4. The Test System responds with a SetChargingProfileResponse with: status <i>Accepted</i> | 3. The CSMS sends a SetChargingProfileRequest <u>Note(s)</u> : - If NotifyEVChargingNeedsResponseStatus was <i>Processing</i> , the Test System will wait 60 seconds for the request |
| 5. The Test System sends a NotifyEVChargingScheduleRequest with evseld <i><COnfigured evseld></i> chargingSchedule <i><ChargingSchedule provided at step 3></i> | 6. The CSMS responds with a NotifyEVChargingScheduleResponse . |
| 7. The Test System sends a TransactionEventRequest with triggerReason <i><ChargingStateChanged></i> transactionInfo.chargingState <i><Charging></i> | 8. The CSMS responds with a TransactionEventResponse . |

| Tool validations |
|--|
| <p>* Step 2: Message: NotifyEVChargingNeedsResponse - Status must be <i>Accepted</i> or <i>Processing</i></p> <p>* Step 3: Message: SetChargingProfileRequest - chargingProfilePurpose must be <i><TxProfile></i> - transactionId must be <i><Provided transactionId from before></i></p> <p>* Step 4: Message: NotifyEVChargingScheduleResponse - status must be <i><Accepted></i></p> |
| N/a |

RenewChargingStationCertificate

| | |
|--------------------------|---|
| State | RenewChargingStationCertificate |
| System under test | CSMS |
| Description | The ChargingStationCertificate is renewed using A02/A03 |

| |
|------------------------------------|
| Before (Preparations) |
| Configuration State: N/a |
| Memory State: N/a |
| Reusable State(s): N/a |

| | |
|---|--|
| Main (Test scenario) | |
| Charging Station | CSMS |
| <u>Manual Action</u> : Request the CSMS to send a Trigger Message Request with requestedMessage SignChargingStationCertificate | |
| 2. The Test System sends a TriggerMessageResponse with status Accepted | 1. The CSMS sends a TriggerMessageRequest With requestedMessage SignChargingStationCertificate |
| 3 The Test System sends a SignCertificateRequest | 4. The CSMS responds with a SignCertificateResponse with status Accepted |
| 6. The Test System sends a CertificateSignedResponse with status Accepted | 5. The CSMS sends a CertificateSignedRequest With certificateChain <Certificate generated from the received CSR from step 3 and signed by the configured CSMS Root certificate> certificateType ChargingStationCertificate |

| |
|--|
| Tool validations |
| <p>* Step 1: Message: TriggerMessageRequest - requestedMessage must be SignChargingStationCertificate</p> <p>* Step 4: Message: SignCertificateResponse - status must be Accepted</p> <p>* Step 5: Message: CertificateSignedRequest - certificateChain <Certificate generated from the received CSR from step 3 and signed by the configured CSMS Root certificate> - certificateType must be ChargingStationCertificate</p> |
| Post scenario validations: N/a |